Advances in Soft and Evolutionary Computing

Research in Computing Science

Series Editorial Board

Research in Computing Science es una publicación trimestral, de circulación internacional, editada por el Centro de Investigación en Computación del IPN, para dar a conocer los avances de investigación científica y desarrollo tecnológico de la comunidad científica internacional. **Volumen 116,** septiembre 2016. Tiraje: 500 ejemplares. *Certificado de Reserva de Derechos al Uso Exclusivo del Título* No. : 04-2005-121611550100-102, expedido por el Instituto Nacional de Derecho de Autor. *Certificado de Licitud de Título* No. 12897, *Certificado de licitud de Contenido* No. 10470, expedidos por la Comisión Calificadora de Publicaciones y Revistas Ilustradas. El contenido de los artículos es responsabilidad exclusiva de sus respectivos autores. Impreso en la Ciudad de México, en los Talleres Gráficos del IPN – Dirección de Publicaciones, Tres Guerras 27, Centro Histórico, México, D.F. Distribuida por el Centro de Investigación en Computación, Av. Juan de Dios Bátiz S/N, Esq. Av. Miguel Othón de Mendizábal, Col. Nueva Industrial Vallejo, C.P. 07738, México, D.F. Tel. 57 29 60 00, ext. 56571.

Editor responsable: *Grigori Sidorov, RFC SIGR651028L69*

Research in Computing Science is published by the Center for Computing Research of IPN. **Volume 116**, September 2016. Printing 500. The authors are responsible for the contents of their articles. Printed in Mexico City, in the IPN Graphic Workshop – Publication Office.

Advances in Soft and Evolutionary Computing

Oscar Herrera Alcántara (ed.)

Instituto Politécnico Nacional, Centro de Investigación en Computación
México 2016

ISSN: 1870-4069

Instituto Politécnico Nacional (IPN)
Centro de Investigación en Computación (CIC)
Av. Juan de Dios Bátiz s/n esq. M. Othón de Mendizábal
Unidad Profesional "Adolfo López Mateos", Zacatenco
07738, México D.F., México

http://www.rcs.cic.ipn.mx
http://www.ipn.mx
http://www.cic.ipn.mx

Indexed in LATINDEX, DBLP and Periodica

Printing: 500

Printed in Mexico

Editorial

This volume of the journal "Research in Computing Science" contains selected papers related to **Soft Computing and in a large part to Evolutionary and Bioinspired Computation**. The papers were carefully selected by the editorial board on the basis of the at least two reviews by members of the reviewing committee or additional reviewers. The reviewers took into account the originality, scientific contribution to the field, soundness and technical quality of the papers. It is worth noting that various papers for this special issue were rejected.

As far as **Evolutionary and Bioinspired Computation** are concerned, the papers of this volume describe the comparison of a Genetic Algorithm and the Population-Based Incremental Learning Algorithm, the analysis of a Particle Swarm Optimization Algorithm using a 3D application, a videogame approach of the Ant Colony Algorithm, and its usage in learning of Bayesian Networks. Also, they concern to the application of the Hybrid Bacterial Foraging Optimization Algorithm for the Cardinality Constrained Portfolio Selection Problem, the improvement of the performance of SVM using Genetic Algorithms, the application of Genetic Programming to image recognition, as well as the use of the LMC complexity measure as a similarity measurement to discover drugs candidates.

As far as **Soft Computing** is concerned, the papers of this volume discuss the usage of Recurrent Neural Networks in the prediction of a tire contact area of a vehicle, the control of an angular-linear position system, an embedded system based in an FPGA for the measurement and processing of electric variables, a fault detection and diagnosis system based on the history data of fault detection and diagnosis of manufacturing systems, the design of an analog-digital processor for Neural Networks implemented in low-power systems, the pattern classification with Spiking Neural Networks, and an approach based on the set theory for similarity metrics with binary attributes.

I like to thank the Mexican Society for Artificial Intelligence (Sociedad Mexicana de Inteligencia Artificial) for its support during the preparation of this volume.

The entire submission, reviewing, and selection process, as well as preparation of the proceedings, were supported for free by the EasyChair system (www.easychair.org).

Oscar Herrera Alcántara
Guest Editor
professor, UAM, Mexico City

September 2016

Table of Contents

Page

Un videojuego educativo basado en la optimización con colonia de hormigas

Abraham Sánchez L., Martin Garcia M., Miguel A. Jara M., José L. Estrada M.

Benemérita Universidad Autónoma de Puebla, Puebla, México

asanchez@cs.buap.mx, yugiharry@me.com, mikejm92@gmail.com, dirus.umbratrox@gmail.com

Resumen. Tradicionalmente, los videojuegos han provisto un marco de estudio para la inteligencia artificial. El principal objetivo de esta investigación es adaptar el algoritmo de la colonia de hormigas, este se basa en el comportamiento natural de las hormigas para buscar una solución a los problemas que requieren adaptabilidad y trabajo cooperativo. Utilizamos las ideas del algoritmo tradicional de optimización con colonia de hormigas en un agente competitivo dentro de un ambiente 3D, implementado en un videojuego de tipo educativo. Las características más importantes del trabajo son la optimización de cálculos para mejorar la fluidez del videojuego y una arquitectura que permite al videojuego servir como educador para aquellos usuarios interesados en conocer los aspectos generales de esta metaheurística.

Palabras clave: Videojuego, colonias de hormigas, optimización, metaheurística.

An Educational Video Game Based on the Ant Colony Optimization

Abstract. Tradicionalmente, los videojuegos han provisto un marco de estudio para la inteligencia artificial. El principal objetivo de esta investigación es adaptar el algoritmo de la colonia de hormigas, este se basa en el comportamiento natural de las hormigas para buscar una solución a los problemas que requieren adaptabilidad y trabajo cooperativo. Utilizamos las ideas del algoritmo tradicional de optimización con colonia de hormigas en un agente competitivo dentro de un ambiente 3D, implementado en un videojuego de tipo educativo. Las características más importantes del trabajo son la optimización de cálculos para mejorar la fluidez del videojuego y una arquitectura que permite al videojuego servir como educador para aquellos usuarios interesados en conocer los aspectos generales de esta metaheurística.

Keywords: Videogame, ant colony, optimization, metaheuristic.

1. Introducción

El proyecto presentado en este trabajo, está inspirado en el algoritmo ACO (del inglés, Ant Colony Optimization) e implementado en un videojuego. El proyecto pretende demostrar la compatibilidad y utilidad entre la inteligencia artificial y los videojuegos [5], así como optimizar el algoritmo mencionado a través de su implementación en un ambiente 3D y la optimización de los cálculos para evitar que el videojuego pierda fluidez [4]. Después de revisar la literatura, es claro que se trata de una idea innovadora, y no se encontraron trabajos relacionados a videojuegos educativos.

En esta propuesta, el videojuego permite al usuario experimentar los cambios del algoritmo dadas ciertas modificaciones en los parámetros y variables utilizadas por el mismo, además de proporcionar al jugador un panorama similar al de las hormigas, a través de un sistema de penumbra que permite al usuario experimentar la sensación de ceguera con la ausencia de la feromona.

La mecánica de la aplicación tiene como objetivo crear un videojuego de estrategia donde el jugador deberá tomar las decisiones correctas para que un imperio de robots prospere. Para cumplir con este objetivo, el usuario debe administrar los recursos que obtengan sus agentes inteligentes (los robots recolectores) para poder optimizar el algoritmo (modificando los parámetros del mismo a través de edificios). Cada optimización tiene un cierto costo de material/alimento, y cada agente consume una cierta cantidad de alimento conforme avanza el tiempo. El algoritmo que se presenta para cada uno de los agentes antes mencionados, es una mejora al algoritmo de optimización con colonia de hormigas; así como una adaptación del mismo a un ambiente tridimensional, donde el tiempo es un elemento muy importante en la toma de decisiones de los robots.

Además, el jugador tendraá una dependencia con la feromona para poder visualizar el mapa; ya que la iluminación del videojuego está basada en la cantidad de feromona contenida en cada cuadrante. Gracias a todos estos elementos, el jugador será capaz de reconocer los efectos que cada parámetro del algoritmo provoca en la recolección de alimentos de los robots.

La aplicación fue implementada utilizando la "plataforma de desarrollo para creación de videojuegos y experiencias interactivas 3D y 2D": Unity 3D en su versión 5 [1], la cual permitió crear el escenario 3D, agregar física a los cuerpos del videojuego, calcular las distancias y tiempos en el ambiente en tiempo real, así como otras funciones secundarias como aumentar la calidad gráfica del juego, etc. El lenguaje de programación utilizado fue C# y las distintas funciones de la aplicación han sido repartidas en múltiples scripts que permiten a todos los elementos funcionar de manera independiente.

En la seccin dos, se presentan algunos aspectos relacionados con el desarrollo de esta propuesta de trabajo. El modelo matemático del algoritmo de la colonia de hormigas y su adaptacin a la propuesta de este trabajo, se presentan en la sección tres. La sección cuatro detallas los elementos más importantes en el desarrollo del videojuegob y finalmente la sección cinco resalta las conclusiones.

2. Desarrollo del algoritmo

El algoritmo de la optimización con colonia de hormigas se inspira en el comportamiento natural de las hormigas en su entorno natural. Las hormigas deambulan libremente dejando una diminuta cantidad de feromona (de forma aleatoria), hasta encontrar algún alimento. Una vez que las hormigas encuentran el alimento, el rastro de feromona se vuelve más intenso; si antes liberaban una pequeña cantidad de feromona, ahora liberan una cantidad constante de feromona. Este comportamiento permite a las hormigas comunicarse entre ellas y seguir rastros de feromona que de alguna forma marcan el camino indicado. Cuando las hormigas que están en un estado de libre recorrido, detectan un rastro de feromona, éstas alteran su comportamiento y comienzan a avanzar guiadas por la feromona; hasta que eventualmente alcanzan el alimento que alguna hormiga probablemente encontró poco tiempo antes.

Después de un cierto tiempo, el alimento se termina, pero las hormigas continúan siguiendo el camino marcado por la feromona. Al desaparecer el alimento, las hormigas que siguen el rastro de feromona rebasan el límite del camino marcado y poco a poco forman su propio camino sin feromona (sin un alimento que frene su avance). Esto provoca que la feromona se evapore lentamente, haciendo que ese camino que ya no conduce a ningún alimento sea menos elegido [2].

Entre más corto sea el camino recorrido por las hormigas, más se concentrará la feromona debido a que las hormigas recorren constantemente un mismo camino y tardan menos tiempo en volver a pasar por el mismo punto, evitando la evaporación de la feromona y fomentando la concurrencia de las hormigas en ese camino. El algoritmo es implementado con un grupo colectivo de agentes inteligentes, que simula el comportamiento de las hormigas, recorriendo un grafo que representa un cierto problema a resolver utilizando mecanismos de cooperación y adaptación [2], [3].

En la implementación de de esta propuesta de trabajo, se sustituyeron las hormigas por robots, representando el problema de forma que estos puedan actualizar sus soluciones, en base a reglas de probabilidad sustentadas en los niveles de feromona de cada camino posible. Además, es importante plantear una heurística σ que mida la calidad de cada dirección que se tomará. Analizando los posibles nodos que podría elegir un robot y seleccionando el más óptimo según la heurística elegida. Dado que la selección de la dirección a tomar está sujeta en todo momento a funciones probabilísticas; la heurística permite definir el camino más probable a seguir, es decir, la selección de caminos es relativamente aleatoria. Finalmente, es necesario definir una regla τ que permita actualizar la feromona.

3. Modelo matemático del algoritmo propuesto

El modelo matemático del algoritmo constituye su elemento más importante. Este se encarga de controlar las iteraciones para realizar las actualizaciones de

Fig. 1. Robot o agente inteligente propuesto para la implementación del algoritmo

los nodos y sus feromonas, lo que permite a los robots elegir la dirección que tomarán. De igual manera, el modelo matemático aporta al jugador la opción de modificar la inteligencia de cada uno de los agentes para encontrar alimento y regresar a la base (a través de sus parámetros). Tradicionalmente, el algoritmo está enfocado en buscar una solución a un cierto problema. Un ejemplo de un problema común es el agente viajero, en donde se busca encontrar el camino más corto de una ciudad a otra a través de un grafo. Para este proyecto, los robots están programados para buscar comida (materiales que les sirven de alimento).

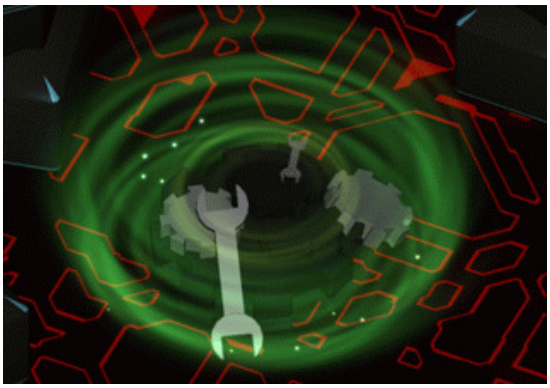


Fig. 2. Alimento que deben encontrar los robots

En el algoritmo tradicional [2], los agentes recorren un grafo donde en cada "turno", cambian de nodo; esto quiere decir que en el algoritmo tradicional la medida de tiempo está basada en turnos y no en mediciones reales de tiempo. La nueva representación del tiempo en el modelo matemático del algoritmo fue la primera aportación de esta investigación; es decir, las actualizaciones se realizan

en tiempo real. Esto significa que los robots están en constante movimiento y no están sometidos a "turnos", sino más bien a cambios de estado. Se propuso utilizar cambios de posición en relación a distancias en metros y poder medir estos cambios de estado (mediante un sí y un no); esto quiere decir que en lugar de un grafo, los robots están orientados por una matriz y en esta matriz cada cuadro está separado por un metro de distancia de su centro al centro del siguiente cuadro. Estos cuadros son coordenadas en x, z dentro de un mundo 3D. Cuando un robot se aproxima a un punto medio de algún cuadrado de la cuadrícula (por ejemplo: (3,4, 1) se aproxima a $(3, 1)$ si el robot que estaba en esa posición avanza a (3,6, 1) el robot se considerará en el cuadro $(4, 1)$ por aproximación), se considera que se ha presentado un cambio de estado y en ese momento se recalcula la fórmula para determinar hasta este punto, que dirección debe tomar el robot(Norte, Noreste, Este, Sureste, Sur, Suroeste, Oeste, Noroeste).

Se examina a continuación la fórmula tradicional para seleccionar el siguiente nodo a visitar:

$$P_{i,j}^{k}(t) = \frac{[\tau_{i,j}(t)]^{\alpha}[\eta_{i,j}(t)]^{\beta}}{\sum_{t \in N_i^k}[\tau_{i,j}(t)]^{\alpha}[\eta_{i,j}(t)]^{\beta}} j \in N_i^k, \tag{1}$$

donde:

- $P_{i,j}^{k}(t)$ la probabilidad de visitar un nodo.
- N_i^k son los nodos no visitados por el robot.
- τ es el momento (en tiempo real) en que se analiza el nodo.
- i coordenada x.
- j coordenada y.
- $\tau_{i,j}(t)$ es el valor de feromona en el punto i, j en un momento t.
- $\eta_{i,j}(t)$ es la heurística a priori que contiene el nodo i, j en un momento t.
- α peso de la feromona, debe ser mayor o igual a 0.
- β peso de la información heurística.

En la investigación presentada, se determinó que la mejor heurística a priori η para el videojuego debía ser la distancia del nodo con respecto al origen, que es la base (donde los robots depositan la comida una vez que la han encontrado). Gracias a esta heurística, los robots pueden encontrar los alimentos más cercanos a la base y, una vez que han obtenido alimento, determinar el camino más corto de regreso a la colonia (a veces ignorando el camino que recorrieron para llegar hasta ese nodo). En este trabajo, el algoritmo original de optimización por colonia de hormigas fue modificado en distintos aspectos; para lograr una adaptación que aumentara la inteligencia de los robots y así mejorar el entretenimiento que el juego aporta. Una de las mejoras más importantes que se debe mencionar con respecto a la fórmula tradicional antes presentada, es que los robots tienen preferencia por ciertas direcciones así como distintos tipos de feromona.

La fórmula $[\tau_{i,j}(t)]^{\alpha}[\eta_{i,j}(t)]^{\beta}$ se modifica de la siguiente manera durante la selección del siguiente nodo a recorrer:

1. Si el nodo analizado se encuentra hacia la misma dirección que el robot recorría, la fórmula cambia a:

$$[\tau_{i,j}(t)]^{\alpha}[\eta_{i,j}(t)]^{\beta} \times 2. \tag{2}$$

2. Si el nodo analizado contiene un camino exitoso que debe recorrer, es decir, se le agrega prioridad a los nodos por donde haya caminado un robot que lleve alimento, la fórmula es:

$$[\tau_{i,j}(t)]^{\alpha}[\eta_{i,j}(t)]^{\beta} \times 15. \tag{3}$$

3. Si el robot carga alimento, la heurística es potenciada y el robot aumenta su preferencia por los nodos cercanos a la colonia, conforme a su valor de inteligencia I (existen edificios que pueden mejorar la inteligencia de los robots):

$$[\tau_{i,j}(t)]^{\alpha}[\eta_{i,j}(t)]^{\beta+I}. \tag{4}$$

4. Finalmente, para seleccionar la siguiente dirección que debe tomar el robot, se realiza un sorteo aleatorio donde la probabilidad (obtenida por la fórmula anterior) de ser un nodo útil, aumenta la posibilidad de tomar determinada dirección. Este sorteo se realiza únicamente entre tres nodos distintos (esto reduce el análisis de nodos y además permite simular de una forma más realista el recorrido de los robots). Por ejemplo: Si un robot camina hacia el norte, los nodos a analizar serán el Noreste, Norte, y Noroeste, por ser los más cercanos al norte.

Los datos anteriores dependen de un valor de feromona τ que se actualiza constantemente. Para obtener este valor de feromona en cada nodo, el método tradicional utiliza la siguiente fórmula y la asigna a cada nodo en cada "turno":

$$\tau_{i,j}(t) = (1-\rho) * \tau_{i,j} + \sum_{k} \triangle {\tau_{i,j}}^{k}, \tag{5}$$

donde:

- ρ es el rango de evaporación de la feromona $0 < \rho < 1$.
- $\sum_k \triangle {\tau_{i,j}}^{k}$ es el cambio de feromona en el nodo y se calcula con la fórmula:

$$\frac{Q}{L_k(t)}, \tag{6}$$

  donde Q es la constante de actualización de la feromona y $L_k(t)$ es el costo en longitud del nodo hacia la base.

Sin embargo, en un ambiente como el de un videojuego donde las operaciones se deben realizar constantemente; calcular todos los valores de feromona en cada uno de los nodos del nivel, es muy costoso en recursos computacionales. Es por ello que en esta propuesta, se implementa una solución que consiste en calcular el valor de la feromona en el nodo, solo en el momento en que se pretende analizar si el nodo será ocupado o no (es decir, cuando los robots cambien de posición). De esta manera, cada uno de los nodos cuenta con una variable personal que almacena el último momento en que fue evaluada la feromona del nodo, y cuando el robot requiere utilizar el nodo, se calcula el tiempo que ha transcurrido desde

esa actualización, y para simular la evaporación de la feromona, se le somete a una función exponencial expresada con la siguiente fórmula:

$$(1-\rho)^{\text{tiempo actual}-\text{ultima actualización}} * \tau_{i,j} + \frac{Q}{L_k(t)}. \tag{7}$$

Así, la variable de actualización se está relacionada con el tiempo, pero se puede posponer su actualización hasta el momento en que se utilizará. A continuación se detalla la implementación de esta propuesta en un videojuego de tipo educativo.

4. Implementación en el videojuego

El videojuego consiste en asegurar la supervivencia de una base de robots. Estas máquinas, interactúan con su medio ambiente a través del algoritmo de optimización por colonia de hormigas modificado. Las máquinas se pueden mejorar buscando algunos edificios que se encuentran en el juego. El jugador debe cuidar su almacenamiento de alimento, así como proteger a los robots de las patrullas enemigas que aparecerán eventualmente. Cada robot evalua la mejora presentada del algoritmo, y los valores de feromona dentro de la matriz de nodos, será un factor crucial en el juego, debido a que el sistema de penumbra impedirá que el jugador pueda ver con claridad, a menos que los robots esparzan su feromona por el mapa, activando las luces dentro del mismo.

4.1. Controles

Debido al objetivo y al gameplay del juego, el videojuego se maneja prácticamente a través de botones. Sin embargo, el jugador puede utilizar distintos controles para interactuar con la aplicación, mejorando la comunicación entre el humano y el software y garantizando que el juego sea más divertido.

- **Cámara**. La cámara del videojuego es totalmente controlable, esta nos permite explorar el mapa sin problemas, moviéndonos hacia arriba, izquierda, abajo y derecha con los botones W, A, S, y D sucesivamente, o bien con los botones de dirección del teclado (botones de flecha). Además, para poder apreciar con detalle los elementos del videojuego, se ha implementado la función de acercamiento y alejamiento de la cámara. Dicha función es manejada con los botones + para acercar la cámara y − para alejar la cámara. A partir de cierta distancia de cercanía, la cámara comenzará a rotar para ofrecer una perspectiva distinta del juego, dicha rotación será nulificada conforme se aleje la cámara de regreso a su posición original o de forma más lejana.
- **Borrado de la feromona**. El botón derecho del ratón permite mostrar y borrar caminos de feromona, posicionando el apuntador sobre una de las esferas azules y manteniendo el botón derecho oprimido.

4.2. Sistema de penumbra

Al inicio de cada ejecución, el mapa está completamente obscuro y el jugador no puede ver más que la base y sus robots. Conforme los robots esparcen feromona por los nodos que recorren, una serie de luces comienzan a encenderse y poco a poco el mapa es más visible. El sistema de penumbra provee al mapa de una luz cada 20 nodos (metros), dentro de cada cuadrante se realiza un promedio de feromona, la cual decidirá la intensidad con la que se enciende cada luz. Si la feromona corresponde a un camino de regreso de los robots (las esferas azules), la iluminación del cuadrante será mucho más rápida, ver la siguiente figura.

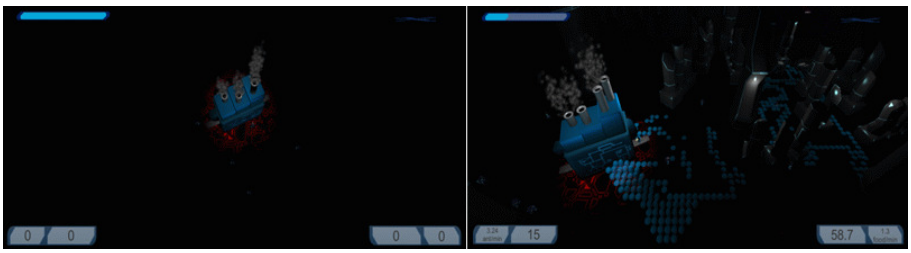


Fig. 3. Al inicio del juego, el jugador se encuentra en penumbras y conforme avanza en el juego, las luces adoptan la forma de la feromona

4.3. Enemigos

Los robots se enfrentan a diversos peligros como la inanición (cuando no hay alimento, los robots comienzan a morir); una esquina donde se encuentra una patrulla que destruye a los robots; y conforme la base crezca y el número de robots aumente, habrá una invasión de pequeñas patrullas que perseguirán a los robots por todo el mapa.

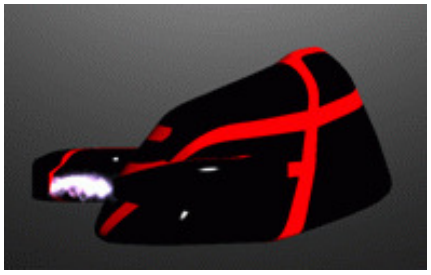


Fig. 4. La patrulla persigue y destruye a los robots

4.4. Edificios

Para realizar mejoras a los robots, el jugador tiene la oportunidad de comprar edificios pagando una cierta cantidad de comida. Una vez construidos, estos e-dificios aportan ciertas ventajas al juego, como protección o perfeccionamiento de la inteligencia y la capacidad de supervivencia de los robots.

En total, el juego cuenta con cuatro diferentes edificios, cada uno con su habilidad y apariencia propia. Uno de estos cuatro edificios (la colonia) se construye cada vez que se inicia el juego, por lo tanto el jugador no debe gastar comida para construirlo. Los cuatro edificios son:

- **Base**: La base es la piedra angular del videojuego, sin ella no sería posible el funcionamiento del videojuego. La base realiza el mapeo del escenario en una matriz de $N \times M$ y guarda los valores de la feromona para cada nodo. Además, gracias a la base es posible mostrar la feromona como objetos transparentes de color azul, facilitando el seguimiento y el borrado de la misma por parte del jugador. Otra de las funciones de la base, es la generación de los robots en un cierto tiempo y el cumplimiento de ciertas condiciones (el tiempo de espera, la cantidad de comida necesaria, etc.).
  La base también se encarga de realizar el conteo de robots y de comida que se visualizan en la Interfaz General del Usuario. Solo es posible tener una base en el videojuego.

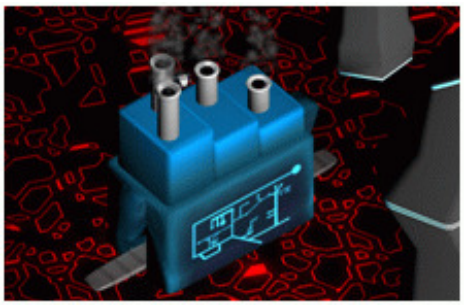


Fig. 5. La base de los robots, donde estos deberán depositar el alimento

- **Torres**: Uno de los mayores retos del videojuego es asegurar la supervivencia de la colonia de los diversos enemigos que rondan a esta. Para protegerse de las patrullas que intentarán destruir a los robots, el jugador puede posicionar torres de defensa que atacarán a las patrullas periódicamente, si estas se acercan lo suficientemente. No existe un límite para colocar torres; sin embargo el jugador debe tener presente que el espacio para colocar edificios es limitado, por lo que deberá colocar todas las torres de una manera estratégica.
- **Máquinas de engranes**: La máquina de engranes es una fábrica que con el paso del tiempo crea engranes. Estos pueden ser utilizados para incrementar

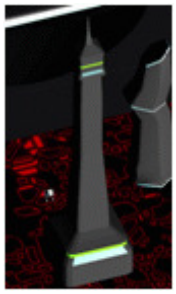


Fig. 6. La torre de defensa permite atacar a las patrullas

la producción de alimento y así favorecer el crecimiento de la base, así como también la recolección de esta materia prima tan importante. Gracias a la máquina de engranes, es posible mejorar la cantidad de alimento aportada por cada engrane; para lograr esto, es necesario hacer clic en el edificio para desplegar el menú de mejoras, y posteriormente utilizar el botón de mejora, siempre y cuando se cuente con la cantidad de alimento requerida para realizar dicha mejora. Cabe mencionar que al igual que las torres, se pueden crear varias instancias de la máquina de engranes si se colocan con cuidado alrededor del mapa.

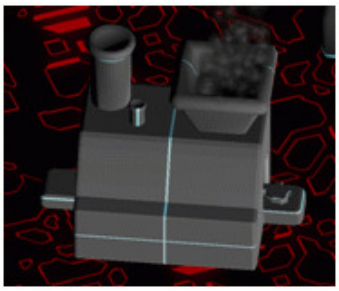


Fig. 7. La máquina de engranes permite crear alimento para los robots

- **Gimnasio**: El gimnasio es un edificio muy útil si se desea optimizar la eficiencia de los robots que están por nacer. Al hacer clic en este edificio, podemos desplegar un menú que ofrece diversas mejoras para los robots. Las mejoras aumentan su precio conforme son compradas, lo que quiere decir que entre más optimizados estén los robots, será más costoso adquirir alimento para la siguiente mejora. Las mejoras que permite realizar el gimnasio son:
  - Duración de la feromona.
  - Velocidad del robot: De 600 inicial hasta 9000 final.
  - Vida del robot.

- Fuerza del robot.
- Inteligencia del robot: De 15 inicial hasta 25 final (sólo afecta a los robots que ya han obtenido alimento y regresan a la base)

En el caso del gimnasio, solo se permite tener una instancia del edificio.

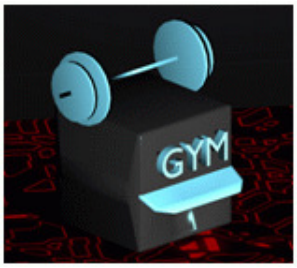


Fig. 8. El gimnasio permite mejorar los parámetros del algoritmo

4.5. Colocación de edificios

Una vez que se ha juntado la cantidad necesaria para crear uno de los edificios disponibles, es posible comenzar la etapa de construcción del edificio. Esta etapa consiste en buscar el lugar adecuado para colocar el edificio. Para evitar abusos en la colocación de edificios, los edificios tienen un límite de distancia entre cada uno, lo cual quiere decir que no es posible colocar dos edificios totalmente juntos o encimados.

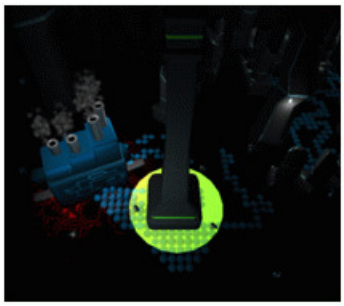


Fig. 9. Una posición válida para colocar un edificio

Durante la etapa de colocación, el usuario deberá mover el ratón a través del escenario; el edificio que se va a construir aparecerá en el mismo lugar en el que se encuentra el apuntador del ratón, así como un objeto que le permitirá saber si la posición es válida o no. Este objeto cambia de color según la proximidad del

futuro edificio con los edificios ya colocados, se pondrá de color rojo si la posición es inválida y de color verde si la posición es válida. Una vez que el usuario esté satisfecho con el lugar de posición, se puede proceder a hacer clic izquierdo para colocar el edificio en su lugar.

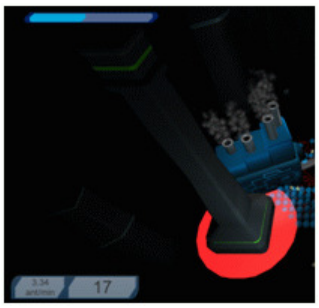


Fig. 10. Una posición inválida para colocar un edificio

La siguiente figura muestra una instantánea del video juego propuesto en este trabajo.

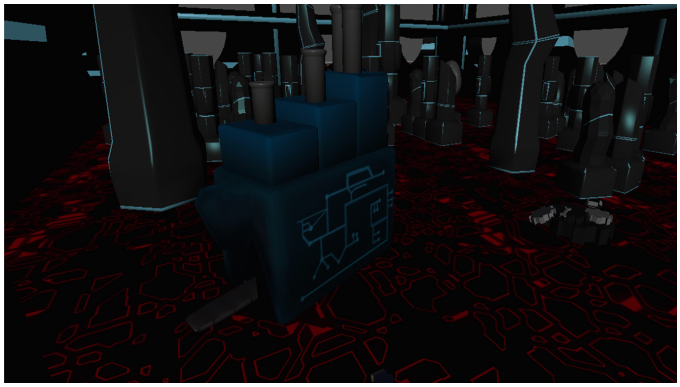


Fig. 11. Ejemplo del video juego propuesto

5. Conclusiones

El algoritmo de optimización con colonia de hormigas tradicional fue adaptado de tal forma, que se puede afirmar que la similitud del comportamiento natural de las hormigas es verdaderamente sorprendente. Estos comportamientos son totalmente plausibles y visualizados en un videojuego 3D. Incluso, el jugador

no es capaz de observar si hay o no feromona en el videojuego. Lo que nos lleva a afirmar que la adaptación se ha llevado de forma satisfactoria al videjuego. Incluso podríamos aventuranos a afirmar que se han hecho ciertas mejoras, obviamente al caso de su adaptación a videojuegos.

Para la realización de este proyecto fue necesario utilizar las tres reglas del algoritmo clásico:

1. Definir correctamente el problema: Conseguir la mayor cantidad de comida posible y regresarla a la base lo antes posible.
2. Plantear la heurística σ: Buscar los nodos más próximos a la base.
3. Definir una regla τ que permita actualizar la feromona: Aportación de la fórmula exponencial para calcular el tiempo transcurrido entre una y otra actualización de la feromona.

Gracias a estos tres puntos, fue más sencillo definir las reglas del videojuego: Conseguir la mayor cantidad de comida en el menor tiempo posible para hacer crecer la base (en número de robots).

El factor tiempo al principio significó una gran limitante, ya que al actua-lizar en todo momento los valores de la feromona, el video juego desarrollado consumía una gran cantidad de recursos de cómputo y era muy complicado utilizar el videojuego (por su relativa lentitud). Más tarde, esta limitante se convirtió en la regla de actualización de la feromona y esto pudo mantenerla dentro de un margen controlado. Se formuló un esquema dentro del cual, el consumo de la feromona depende exponencialmente del tiempo que ha transcurrido desde la última actualización de la feromona (los nodos no son necesariamente actualizados a cada momento).

La combinación de la inteligencia artificial con videojuegos ha demostrado que ambos son completamente compatibles con la observación de que es importante cuidar el número de cálculos realizados al mismo tiempo ya que esto puede retrasar el tiempo de respuesta del videojuego, y aportar una experiencia no deseable para el jugador.

Referencias

1. Unity Technologies: The best development platform for creating games. Obtained from: http://unity3d.com/unity, (2015)
2. M. Dorigo, T. Stützle: Ant colony optimization. Bradford Books, First Edition (2004)
3. E. Martin, M. Martinez, G. Recio, Y. Saez: Pac-mAnt: Optimization based on ant colonies applied to developing an agent for Ms. Pac-Man. In: IEEE Symposium on Computational Intelligence and Games (CIG), pp. 458–464 (2010)
4. G. N. Yannakakis: Game AI revisited. In: Proc. of ACM Computing Frontiers Conference, pp. 285–292 (2012)
5. Z. A. Algfoor, M. S. Sunar, H. Kolivand: A comprehensive study on path-finding techniques for robotics and video games. International Journal of Computer Games Technology, Vol. 2015 (2015)

Uso del algoritmo de colonia de hormigas en el aprendizaje de redes bayesianas

Guillermo Ramos F., Abraham Sánchez L., Fabian Aguilar C.,
María B. Bernábe L., Rogelio González V.

Benemérita Universidad Autónoma de Puebla,
Computer Science Department,
México

sase.ramses@gmail.com, asanchez@cs.buap.mx, slipknot_1964@live.com.mx,
beatriz.bernabe@gmail.com, rgonzalez@cs.buap.mx

Resumen. Las redes bayesianas modelan un fenómeno mediante un conjunto de variables y las relaciones de dependencia entre ellas. Dado este modelo, se puede hacer inferencia bayesiana. Estos modelos pueden tener diversas aplicaciones, para clasificación, predicción, diagnóstico, etc. Además, pueden dar información interesante en cuanto a cómo se relacionan las variables, las cuales pueden ser interpretadas en ocasiones como relaciones de causa-efecto. El problema radica en que la obtención de la estructura de una red es un problema NP-Duro. En este trabajo se propone un algoritmo para el aprendizaje en redes bayesianas basado en una metaheurística que ha sido aplicada con éxito, la optimización de la colonia de hormigas. Se presentan varios ejemplos para validar nuestra propuesta y comparar en tiempo y clasificación dichos resultados con el algoritmo clásico K2 y el software GeNIe.

Palabras clave: Redes bayesianas, inferencia, aprendizaje, colonia de hormigas.

The Use of the Ant Colony Algorithm in the Learning of Bayesian Networks

Abstract. The bayesian networks model phenomena through a set of variables and their dependence. With these models is possible to develop bayesian inference. The models have several applications in data classification, data prediction, diagnostics, etc. Also, these models provide revelant information about the relationship between the variables and sometimes can be interpreted as a cause-effect relationship. The underlying problem is that the determination of the network structure is an NP-Hard. In this work an algorithm for learning in bayesian networks have been successfuly applied based on the ant colony metaheuristic. Several examples are presented to validate the proposed method and the results of time and classification are compared with those of the traditional algorithm $K2$ and the GeNIe software.

Keywords: Bayesian networks, inference, learning, ant colony.

1. Introducción

Las redes bayesianas (RBs), también conocidas como redes de creencias probabilísticas o redes causales, son herramientas de representación de conocimiento capaces de gestionar eficazmente las relaciones de dependencia/independencia entre las variables aleatorias que componen el dominio del problema que queremos modelar. Estas tienen dos componentes: a) una estructura gráfica, o más precisamente un grafo acíclico dirigido (DAG), y b) un conjunto de parámetros, que en conjunto especifican una distribución de probabilidad conjunta sobre las variables aleatorias.

Los parámetros son un conjunto de medidas de probabilidad condicional, que dan forma a las relaciones. Por esta razón, las RBs tienen muchas ventajas, por ejemplo: logran presentar las interrelaciones de los elementos como un todo, y no sólo por sus partes (por su representación multivariable; tratan el problema del ruido de los datos experimentales), describen las complejas relaciones de los elementos con naturaleza probabilística y no lineal, representan las relaciones causales de las interacciones y manejan eventos que no han sido observados, y la incertidumbre inherente a ellos, etc.

El problema es obtener la estructura y parámetros de una RB a partir de un conjunto de muestras de los elementos de la red. La dificultad radica en el número de posibles redes bayesianas en el espacio de búsqueda que crece en forma exponencial, este problema se define como NP-duro [1].

De forma general, podemos decir que el problema del aprendizaje consiste en construir, partir de un conjunto de datos, el modelo que mejor represente la realidad, mejor dicho, una porción del mundo real en la cual estamos interesados. Como en el caso de la construcción manual de redes bayesianas, el aprendizaje de este tipo de modelos tiene dos aspectos: a) el aprendizaje paramétrico, y b) el aprendizaje estructural [3], [2]. A continuación detallamos algunos conceptos que nos serviran como base a nuestra propuesta.

El presente trabajo se centra en el aprendizaje estructural de una RB en tiempo polinomial, mediante la metaheuristica de la colonia de hormigas (del inglés ACO, Ant Colony Optimization). Una vez determinada la red se compara su efectividad al momento de clasificar. Posteriormente se describe el algoritmo planteado, y para finalizar este trabajo se muestran los resultados de las RB obtenidas en tres ejemplos distintos, así como el tiempo que tomo decretarlas y su efectividad al momento de clasificar datos; comparando dichos resultados con el clásico algoritmo K2 y el programa Genie & smile.

2. Conceptos básicos

En esta sección se revisan brevemente algunos conceptos básicos relacionados con la RB y cómo aprender de ellos, así como otros conceptos relacionados con ACO [4].

2.1. Redes bayesianas

Las siguientes definiciones proveen un marco adecuado para describir en detalle, lo que es una RB desde la definición de un dígrafo dirigido acíclico (del inglés DAG, Directed Acyclic Graph), hasta la distribución de probabilidad que la red representa, mostrando su cualidad principal de independencia condicional.

El aprendizaje de la estructura de una RB se centraliza en la búsqueda de una estructura óptima para un conjunto de datos cuando se escoge el mejor puntaje entre esta; a esto se le conoce como aprendizaje basado en la búsqueda.

La red bayesiana es un DAG $G = (V, E)$, donde el conjunto de nodos "V" representará a nuestras variables de nuestro sistema $(X_1, X_2, ..., X_n)$, "E" como el conjunto de arcos, estas son las relaciones directas entre las variables. Para cada variable $x_i \in V$ tenemos una familia de distribuciones condicionales $P(x_i|Pa(x_i))$, donde $Pa(x_i)$ representa el conjunto de los padres de la variable x_i. A partir de estas distribuciones condicionales podemos recuperar la distribución conjunta sobre V:

$$P(x_1, x_2, ..., x_n) = \prod_{i=1}^{n} P(x_i|Pa(x_i)). \tag{1}$$

Esta fórmula muestra la distribución conjunta por medio de la presencia o ausencia de nuestras conexiones entre pares de las variables. Una propiedad deseable e importante de una métrica es la forma en cómo se descompone en la presencia de datos completos, es decir, la función de puntuación se puede descomponer de la siguiente manera:

$$f(G : D) = \sum_{i=1}^{n} f(x_i, Pa(x_i) : N_{x_i,pa(x_i)}, \tag{2}$$

donde $N_{x_i,pa(x_i)}$ son las estadísticas de la variable x_i y $Pa(x_i)$ en D, es decir, el número de instancias en D que coincidan con cada posible creación de instancias de x_i y $Pa(x_i)$. En este caso se busca un procedimiento capaz de realizar una búsqueda local con la cual podamos cambiar la posición de los arcos para evaluarlos de una manera eficiente y reutilizar la mayor parte de los cálculos.

2.2. Algoritmo B

El algoritmo B es una heurística de construcción voraz, la cual en cada paso agrega un arco con el máximo aumento en el puntaje en la métrica f, pero evitando la inclusión de ciclos dirigidos [5]. El algoritmo termina cuando la adición de cualquier arco no incrementa el valor de la métrica.

2.3. Métrica K2

Entre las distintas técnicas para poder construir una red bayesiana a partir de sus datos, utilizamos el algoritmo K2, ya que este está basado en la optimización

1. INICIALIZACION
a) **for** $i = 1$ **to** n **do**
$Pa(x_i) = \emptyset$
b) **for** $i = 1$ and $j = 1$ **to** n **do**
if $(i \neq j)$ **then** $A[i, j] = f(x_i, x_j) - f(x_i, \emptyset)$
2. CICLO:
Repeat
i. Selecciona dos indices (i, j) tal que
$(i, j) = \arg \max_{i',j'} A[i', j']$
ii. **if** $A[i, j] > 0$ **then** $Pa(x_i) = Pa(x_i) \cup \{x_j\}$
iii. $A[i, j] = -\infty$
iv. **for all** $x_a \in$ Ancestros$(x_j) \cup \{x_j\}$ **and** $x_b \in$ Descendientes$(x_i) \cup \{x_i\}$ **do**
$A[a, b] = -\infty$
v. **for** $k = 1$ **to** n **do**
if $(A[i, k] > -\infty)$ then $A[i, k] = f(x_i, Pa(x_i) \cup \{x_k\}) - f(x_i, Pa(x_i))$
until $\forall i, j$ $(A[i, j] \leq 0$ or $A[i, j] = -\infty)$

Fig. 1. Algoritmo B

de una medida, y esta a su vez es utilizada para la exploración en el espacio formando las redes que contienen nuestras variables en la base de datos [6]. Parte de un DAG y va añadiendo arcos, modificándolos o eliminándolos para poder obtener una red con la mejor medida. Para una red G y una base de datos D, su medida K2 es la siguiente:

$$f(G : D) = \log P(G) + \sum_{i=1}^{n} \left[\sum_{k=1}^{si} \left[\log \frac{\Gamma(\eta_{ik})}{\Gamma(N_{ik}, \eta_{ik})} + \sum_{j=1}^{ri} \log \frac{\Gamma(N_{ik}, \eta_{ik})}{\eta_{ik}} \right] \right], \quad (3)$$

donde:

- N_{ik} es la frecuencia de las configuraciones encontradas en la base de datos D de las variables x_i.
- n es el número de variables, tomando su j-ésimo valor y sus padres en G tomando su k-ésima configuración, donde si es el número de configuraciones posibles del conjunto de padres.
- Γ_i es el número de valores que puede tomar la variable x_i.
- Además, $N_{ik} = \sum_{j=1}^{\Gamma_i} N_{ik}$.
- Γ es la función Gamma.
- El parámetro η puede interpretarse como el tamaño muestral equivalente.

La métrica ha adoptado el nombre del algoritmo, de modo que se conoce como la métrica K2, es decir:

Suponiendo una distribución uniforme a priori de $P(G)$ y usando $log(P(G))$ instanciado de $P(G, D)$, tenemos una métrica descomponible:

$$f_{K2}(G, D) = \sum_{i=1}^{n} f_{K2}(x_i, pa(x_i) : N_{x_i, pa(x_i)}), \tag{4}$$

$$f_{K2}(x_i, pa(x_i) : N_{x_i, pa(x_i)}) = \sum_{j=1}^{qi} \left(\log \left(\frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \right) + \sum_{k=1}^{ri} \log(N_{jnk}!) \right). \tag{5}$$

2.4. Optimización basada en colonia de hormigas

El algoritmo ACO está basado en como las hormigas reales realizan su búsqueda para encontrar comida o un nuevo hogar, ellas deciden cual es el camino más corto para poder encontrar el recurso que necesitan, cuando caminan dejan una pequeña sustancia en el suelo denominada "feromona" [7]. Solo ellas pueden detectar ese aroma y al elegir su camino, tienen que escoger de una manera probabilística los caminos que están marcados con más concentración de feromonas [8].

Cuando no hay feromonas en el lugar, escogen un camino al azar, pero después de un periodo transcurrido, los caminos más cortos serán frecuentados más rápido y a su vez, estos caminos tendrán más feromonas, y esto hace posible que las hormigas utilicen dichos caminos. Este efecto significa que las hormigas han encontrado el camino más corto, es decir, aunque una sola hormiga llegue a la solución, la solución óptima es el resultado del comportamiento cooperativo de las hormigas. Cada hormiga representa una posible solución al problema desplazándose a través de una secuencia finita de nodos, los movimientos que realizan son seleccionados aplicando una búsqueda local, resolviendo los problemas específicos de información local y de la información compartida sobre la feromona [7].

A la feromona la modelamos mediante una matriz τ, donde τ_{ij} contiene el nivel de feromona depositada en el arco del nodo i al nodo j. En los primeros sistemas de hormigas, una hormiga k en el nodo i seleccionará el siguiente nodo j para visitar con la probabilidad:

$$p_k(i, j) = \begin{cases} \frac{[\tau_{ij}]^{\alpha}[n_{ij}]^{\beta}}{\sum_{u \in j_k(i)}}, & \text{si } j \in j_k(i) \\ 0, & \text{en otro caso,} \end{cases} \tag{6}$$

donde n_{ij} representa la información heurística sobre el problema, $j_k(i)$ es el conjunto de nodos vecinos del nodo i que aún no han sido visitados por la hormiga k, α y β son dos parámetros que determinan la importancia relativa de la feromona con respecto a la información heurística.

En cada iteración del algoritmo de cada hormiga, utilizando la regla de transición anterior, se construye progresivamente una solución. La matriz de la feromona se actualiza de la siguiente manera:

- Actualización global: Sólo la hormiga, que construyó la mejor solución refuerza el nivel de feromona en los arcos que forman parte de la mejor solución S^+ obtenida hasta el momento. Esto dirige la búsqueda en el vecindario de la mejor solución.
- Actualización local: Al construir una solución, si una hormiga realiza la transición del nodo i al nodo j, entonces el nivel de feromonas del arco correspondiente se modifica. Esta regla favorece la exploración de otros arcos, evitando así la convergencia prematura; sin la actualización local de todas las hormigas que buscarían en todo el vecindario la mejor solución encontrada hasta el momento.
- Uso de un optimizador local: algunas o todas de las soluciones obtenidas por las hormigas están optimizadas a nivel local mediante el uso de un método de búsqueda local. Esta técnica es particularmente útil para muchos problemas de optimización combinatoria, donde en la práctica se obtienen mejores resultados cuando se acoplan este tipo de algoritmos con optimizadores locales.

3. Redes de aprendizaje bayesiano con colonia de hormigas

En esta sección mostraremos la definición de los componentes que necesitamos para poder aplicar la metaheurística ACO en nuestro problema:

- Representación adecuada del problema: Significa poder construir las posibles soluciones utilizando una regla de probabilidad para poder pasar de un estado i a un estado j.
- Información heurística: Conocimientos específicos que utilizan el proceso de la búsqueda n_{ij}, es decir cuando nos movemos del estado i al estado j.
- Regla(s) para actualizar la matriz de feromonas τ.
- Regla de transición probabilística que utiliza la heurística n y la feromona τ.
- Optimizador local.

Ahora, vamos a definir todos los componentes para nuestro problema de aprendizaje:

- Representación del problema: La representación del problema es un grafo donde los estados del problema son los DAGs con n nodos. Por lo tanto, un estado G_h será un grafo con los nodos $x_i \in V$ y exactamente h arcos y ningún ciclo dirigido. La construcción incremental de la hormiga en la solución se inicia desde el grafo vacío G_0 (arcs-less dag) y procede mediante la adición de un arco $x_j \to x_i$ para el estado actual G_h, es decir, $G_{h+1} = G_h \cup \{x_j \to x_i\}$. La solución final será el estado G_h en el que la hormiga decide dejar la fase de construcción.
- Información heurística:

$$n_{ij} = f(x_i, Pa(x_i) \cup \{x_j\}) - f(x_i, Pa(x_i)).$$

- Reglas de actualización de feromonas: Las reglas globales y locales de actualización se consideran las mismas que se han descrito anteriormente.
- Regla de transición probabilística: El siguiente arco va a ser incluido en el grafo actual G, por una hormiga seleccionada de una manera similar a la utilizada por el algoritmo B, pero utilizando una regla de decisión estocstica (en lugar de una regla determinista) que también tiene en cuenta la feromona depositada en cada arco.
- Optimizador local: Es una mejora donde, en cada paso, el mejor movimiento de acuerdo con la métrica y los operadores utilizados se seleccionan. La complejidad de estos movimientos es $O(n^2)$, donde n es el número de variables. Debemos tener en cuenta que si se utiliza una métrica descomponible, un gran número de cálculos puede ser reutilizado desde las etapas anteriores del algoritmo. También hay que tomar en cuenta que los operadores de transición elegidos contienen el elegido para la hormiga B. Por lo tanto, una vez que una hormiga ha obtenido una solución, entonces, mediante la supresión o inversión de un arco, el algoritmo puede escapar de un óptimo local eventual alcanzado por la hormiga.

4. Resultados experimentales

Para comparar los resultados se utilizaron tres conjuntos de datos disponibles en UCI Machine Learning Repository y se evaluaron las estructuras obtenidas por ACO con respecto al algoritmo K2 y la herramienta GeNIe & Smile de la Universidad de Pittsburgh. Se utilizaron los siguientes parámetros, $\alpha = 1$, $\beta = 2{,}0$, 10 hormigas y 20 iteraciones.

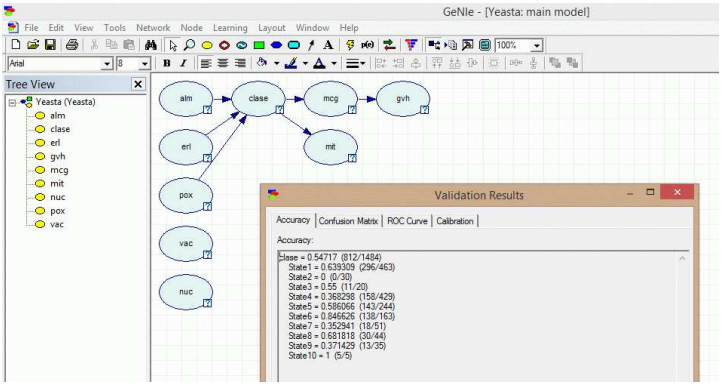
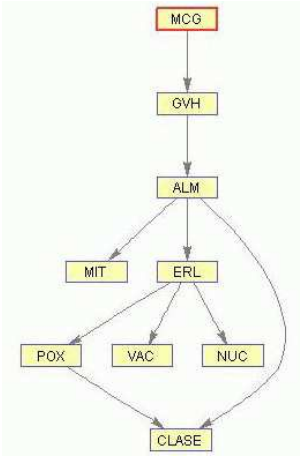


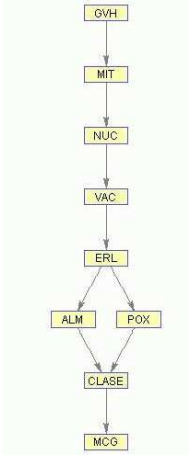
Fig. 2. RB para el ejemplo Yeast utilizando el software GeNIe

En los casos correspondientes donde hay registros incompletos, estos fueron llenados por medio de interpolación, no se utilizaron otras técnicas de preprocesamiento.



Clase	0.415094	616/1484
Estado 1	0.704104	326/463
Estado 2	0	0/30
Estado 3	0.55	11/20
Estado 4	0.317016	136/429
Estado 5	0.00409836	1/244
Estado 6	0.871166	142/163
Estado 7	0	0/51
Estado 8	0	0/44
Estado 9	0	0/35
Estado 10	0	0/5

Fig. 3. RB para el ejemplo Yeast utilizando el algoritmo K2



Clase	0.472372	701/1484
Estado 1	0.7473	346/463
Estado 2	0	0/30
Estado 3	0.55	11/20
Estado 4	0.314685	135/429
Estado 5	0.0409836	10/244
Estado 6	0.858896	140/163
Estado 7	0.25902	13/51
Estado 8	0.613636	27/44
Estado 9	0.542857	19/35
Estado 10	0	0/5

Fig. 4. RB para el ejemplo Yeast utilizando el algoritmo ACO

Las siguientes figuras muestran las RBs resultantes con la herramienta GeNIe y con los dos algoritmos propuestos, K2 y el algoritmo ACO.

En el primer conjunto de datos de nombre Yeast, obtenemos una clasificación baja por parte de las 3 herramientas, donde GeNIe obtiene el mejor resultado con casi 55 %, y le siguen ACO con 47 % y K2 con 41,5 %, ver las figuras 2, 3 y 4. Con tiempos de 1.83, 6.2 y 5.7 segundos respectivamente.

Para el conjunto de datos Mammographic Mass los resultados para GeNIe, ACO y K2 fueron 84,6 %, 83,6 % y 83,5 % respectivamente, ver las figuras 5, 6 y 7. Con tiempos de 1.03, 5.9 y 5.45 segundos respectivamente.

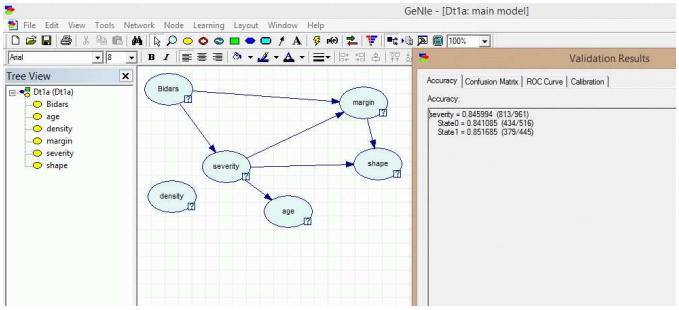


Fig. 5. RB para el ejemplo Mammographic mass utilizando el software GeNIe

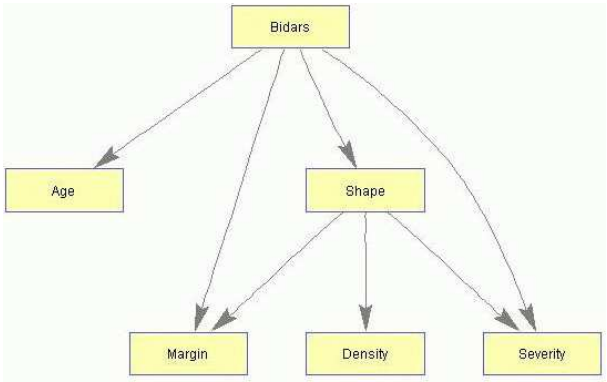


Fig. 6. RB para el ejemplo Mammographic mass utilizando el algoritmo K2

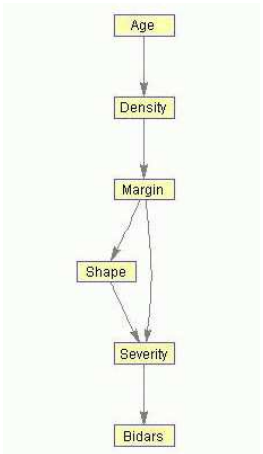


Fig. 7. RB para el ejemplo Mammographic mass utilizando el algoritmo ACO

Para el conjunto de datos Banknote Authentication los resultados para GeNIe, ACO y K2 fueron $95,6\,\%, 89{,}3\,\% y 82\,\%$ respectivamente, ver las figuras 8, 9 y 10. Con tiempos de 1.16, 5.78 y 5.14 segundos respectivamente.

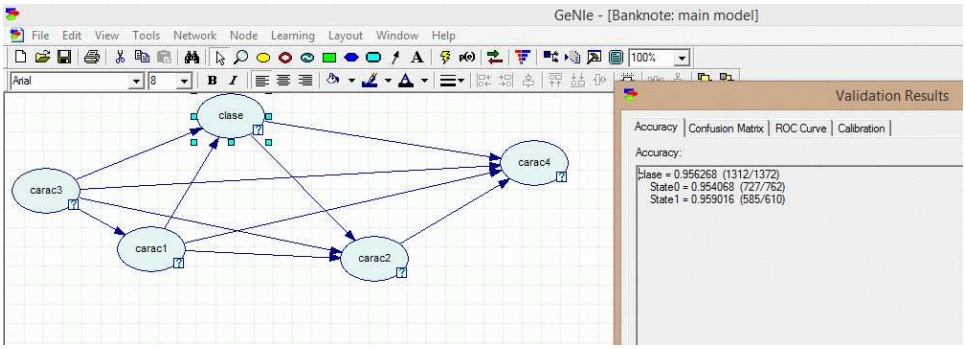


Fig. 8. RB para el ejemplo Banknote Authentication utilizando el software GeNIe

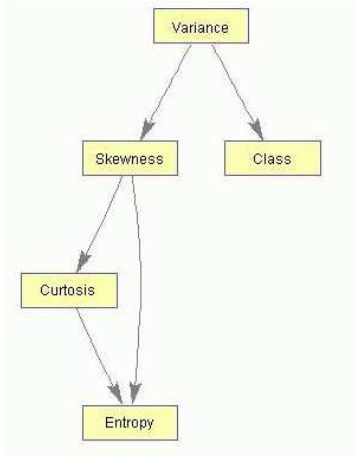


Fig. 9. RB para el ejemplo Banknote Authentication utilizando el algoritmo K2

Las estructuras de las RBs fueron obtenidas por las 3 propuestas (herramienta GeNIe, algoritmo K2 y algoritmo ACO) en pocos segundos, que es el objetivo principal de esta investigación.

Podemos concluir que nuestro algoritmo alcanzo resultados mejores al algoritmo K2, pero que algunas herramientas superan su desempeño, se espera mejorar el algoritmo para obtener mejores resultados en la obtención de estructuras para RBs.

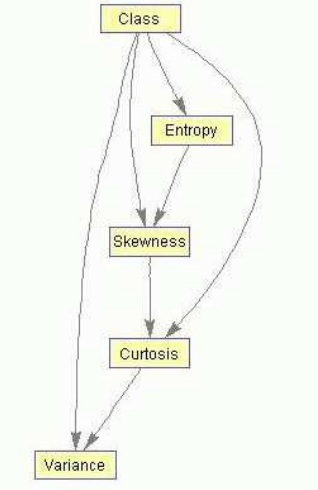


Fig. 10. RB para el ejemplo Banknote Authentication utilizando el algoritmo ACO

5. Conclusiones y trabajo futuro

En este trabajo, obtuvimos diferentes estructuras de una RB dependiendo del algoritmo utilizado y del software, ya que cada software hace de forma diferente la inferencia de los datos y algunos suelen ser mejores que otros, podemos concluir lo siguiente: Hay un buen rendimiento y logramos el objetivo, que fue evitar la búsqueda redundante e ineficiente de hacer todas las permutaciones posibles para obtener la red óptima con el algoritmo de la colonia de hormigas, mientras que en el K2, hay que darle el orden de cómo debe hacer la inferencia en la red.

De acuerdo con lo anterior, se plantea realizar, en trabajos futuros, el aprendizaje de la Red bayesiana, utilizando como base el algoritmo de Colonia de Hormigas para reducir el número de iteraciones en un espacio de búsqueda según el algoritmo K2 y el uso de operadores para moverse entre clases de equivalencia de acuerdo con el algoritmo ACO-B [9].

Referencias

1. Cooper, G. F., Herskovits, E.: A Bayesian method for the induction of probabilistic networks from data. Machine Learning, Vol. 9, pp. 309–348 (1992)
2. Spirtes, P., Meek, C.: Learning Bayesian networks with discrete variables from data. In: Proc. of the First International Conference on Knowledge Discovery and Data Mining, pp. 294–300 (1995)
3. Chickering, D. M.: Search operators for learning equivalence classes of Bayesian network structures. Technical Report, R231, UCLA Cognitive Systems Laboratory (1995)
4. de Campos, L. M., Fernández L., J. M. , Gámez, J. A., Puerta, J. M.: Ant colony optimization for learning Bayesian networks. International Journal of Approximate Reasoning, Vol. 31, pp. 291–311 (2002)

5. Buntine, W.: Theory refinement on Bayesian networks. In: Proceedings of the 7th Conference on Uncertainty in Artificial Intelligence, Morgan Kaufmann, San Mateo, pp. 52–60 (1991)
6. Cooper, G., Herskovits, E.: A Bayesian method for the induction of probabilistic networks from data. Machine Learning, Vol. 9, No. 4, pp. 309-348 (1992)
7. M. Dorigo, T. Stützle: Ant colony optimization, The MIT Press (2004)
8. Dorigo, M., Maniezzo, V., Colorni, A.: The Ant System: Optimization by a Colony of Cooperating Agents. IEEE Trans. on Systems, Man and Cybernetics, Part B, Vol. 26, pp. 29-41 (1996)
9. Puerta C., José M.: Métodos locales y distribuidos para la construcción de redes de creencia estáticas y dinámicas. Tesis de la Universidad de Granada, España (2001)

Detección y diagnóstico de fallas en sistemas complejos de manufactura empleando técnicas de softcomputing

Juan Pablo Nieto González

Universidad Autónoma de Coahuila,
Coahuila, México

juan.nieto@uadec.edu.mx

Resumen. La principal meta de un sistema de detección y diagnóstico de fallas en sistemas de manufactura es prevenir un paro no deseado de la línea de producción y su correspondiente costo. Los modernos sistemas de manufactura son un dominio bastante complejo debido a todas las variables físicas involucradas y a sus correlaciones. Lo cual representa una tarea bastante retadora para los sistemas de monitoreo. El presente artículo muestra un sistema de detección y diagnóstico de fallas basado en los datos históricos del proceso. La propuesta esta compuesta por dos fases. En la primera fase aprende el comportamiento de la operación normal del sistema utilizando una red neuronal autoasociativa (RNAA), la cual lleva a cabo el proceso de detección. En la segunda fase se da el diagnóstico final empleando una máquina de soporte vectorial multiclase (MSV), la cual clasifica el tipo de falla presente y proporciona su tiempo de ocurrencia.

Palabras clave: Detección de fallas, diagnóstico de fallas, sistemas complejos, ed neuronal autoasociativa, máquina de soporte vectorial.

Fault Detection and Diagnostics in Complex Manufacturing Systems using Softcomputing Techniques

Abstract. The main goal of a fault detection and diagnosis of manufacturing systems is to prevent an undesired stop of the production line and its corresponding cost. Modern manufacturing systems are a very complex domain due to all the physical variables involved and their correlations. This represent a very challenging task for monitoring systems. The present article shows a fault detection and diagnosis system based on the history data process. The proposal is composed by two phases. In the first phase it learns the normal operation behavior of the system using an autoassociative neural network which carries out the detection process. In the second phase the final diagnosis is given using

a multiclass support vector machine, which classifies the type of fault present and gives its time of occurrence.

Keywords: Fault detection, fault diagnostics, complex systems, autoassociative neural network, support vector machine.

1. Introducción

Desde el comienzo de la era de las máquinas, los humanos han estado interesados en sus condiciones de trabajo. Durante casi toda la historia de la humanidad, la única manera de aprender acerca de las malas operaciones de los sistemas y sus correspondientes subsistemas, ha sido mediante el empleo de solamente los cinco sentidos. Posteriormente se da la implementación de los sensores, que son elementos destinados a tomar lecturas y son utilizados para dar a los humanos el estado de ciertas variables físicas y de ésta manera detectar si un sistema o proceso está trabajando adecuadamente o no. Con el gran avance tecnológico los procesos se han vuelto cada vez más complejos, por lo cual su monitoreo es bastante importante para mejorar su desempeño, eficiencia y asegurar la calidad del producto terminado. Es entonces, que el análisis y diagnóstico de fallas puede ayudar a evitar pérdidas de producción y accidentes, que ponen en riesgo la salud y vida de los operadores y daño al equipo. El diagnóstico de fallas de los sistemas de ingeniería esta relacionado a la detección de fallas en máquinas complejas detectando patrones específicos de comportamiento en los datos observados. Un moderno sistema de manufactura es un ejemplo de tal sistema de ingeniería complejo dónde existen un gran número de sensores, controladores y módulos de cómputo que recolectan una gran cantidad de señales. Basado en estos hechos y considerando que los modernos sistemas de manufactura industrial, ya sean de una planta industrial completa o una sola máquina, son sistemas de gran escala y de ellos se pueden extraer una gran cantidad de datos, es que se propone una metodología de monitoreo y diagnóstico de fallas que emplea solamente datos históricos del proceso combinando una red neuronal autoasociativa (RNAA) y una máquina de soporte vectorial multiclase (MSV). La metodología se aplica a los motores de una banda transportadora mostrando resultados prometedores. La organización del artículo es la siguiente. La sección 2 revisa el estado del arte. La sección 3 da los preliminares matemáticos de la RNAA y de la MSV. La sección 4 proporciona la descripción general de la propuesta. La sección 5 muestra un caso de estudio. Finalmente la sección 6 da la conclusión del artículo.

2. Estado del arte

El incremento en los requerimientos para alcanzar un desempeño más confiable de sistemas complejos ha motivado el desarrollo de esquemas de diagnóstico de fallas que puedan realizar un diagnóstico más preciso. En el presente

artículo se propone una metodología para el diagnóstico de sistemas complejos. La propuesta fue aplicada al monitoreo de motores eléctricos de una celda de manufactura. Desde el punto de vista de seguridad y confiabilidad de los sistemas eléctricos, es necesario tener un oportuno diagnóstico de fallas que pueda detectar, aislar y diagnósticar fallas y de avisar a los operadores del sistema para tomar las correspondientes acciones correctivas. Durante un disturbio, hay un gran número de eventos relacionados a las fallas, haciendo que el diagnóstico y la decisión de tomar acciones correctivas se torne una tarea difícil. En este dominio, la necesidad de desarrollar técnicas más poderosas ha sido reconocida, y las técnicas híbridas que combinan varios métodos de razonamiento se han empezado a emplear. [1] considera la configuración de elementos automáticos en los modernos sistemas de potencia eléctrica, tales como relevadores de protección y de recierre automáticos para mejorar un modelo analítico y de optimización para el diagnóstico de fallas de sistemas de potencia. De acuerdo al principio de protección de los relevadores, el diagnóstico de fallas es expresado como un problema de programación entera y resuelto por el algoritmo genético de búsqueda Tabú. [2] presenta una metodología que utiliza redes neuronales integradas con varias técnicas estadísticas. Entre las herramientas numéricas y estadísticas utiliza análisis de Fourier, valores RMS, valores de sesgo y de curtosis así como componentes simétricas para la detección y la identificación de las fallas es llevada a cabo mediante una red neuronal perceptrón multicapa. [3] emplea las formas de onda de corriente y les aplican un análisis en frecuencia y onduletas para generar un algoritmo de formación de grupos para clasificar si el sistema se encuentra en modo de falla o no. [4] utiliza lógica difusa para la detección de fallas en motores de inducción. La propuesta esta compuesta por dos etapas. Extracción de razgos y la implementación de un sistema difuso. Se utilizan las magnitudes de las corrientes trifásicas de un motor para detectar y caracterizar las fallas. [5] propone un método compuesto de dos fases: En la primera fase una red neuronal probabilística es entrenada con los eigenvalores de los datos de voltaje obtenidos en operación normal así como con fallas simétricas y asimétricas. La segunda fase emplea una comparación entre las muestras para detectar y localizar la presencia de una falla. [6] presenta una metodología de diagnóstico basada en un sistema difuso que interpreta las señales provenientes de los sensores de corriente de un motor de inducción. [7] encuentra señales de entropía y la curtosis, para posteriormente alimentarlas como entradas de una red neuronal y así llevar a cabo la clasificación de las fallas. [8] utiliza lecturas de la corriente de fase solo durante el primer cuarto de ciclo empleando un método que combina componentes simétricas con un análisis de componentes principales para identificar y clasificar una falla. [9] puede analizar fallas que ocurren entre dos líneas equipadas con unidades de medición. El primer paso de la metodología es detectar la presencia de una falla en tiempo real. Luego, el método de componentes simétricas es utilizado para convertir señales trifásicas en tres conjuntos de componentes independientes, los cuales son las secuencias positiva, negativa y cero. [10] propone una red bayesiana y minería de datos para diagnosticar fallas en una red eléctrica. El estatus de la información de las

protecciones es tomado como atributos condicionales y la región de falla como un atributo de decisión. [11] relaciona la incertidumbre asociada a un evento con una distribución de probabilidad. Utiliza un algoritmo difuso aplicado a cada valor de entropía para realizar la identificación de una falla. [12] propone una metodología capaz de localizar nodos de un sistema eléctrico en modo de falla. La metodología está compuesta por dos fases: en la primer fase una red neuronal probabilística se entrena con los eigenvalores de voltaje para dar una clasificación del tipo de falla presente. En la segunda fase se utiliza un ANFIS para dar el diagnóstico final.

3. Preliminares

3.1. Red neuronal autoasociativa (RNAA)

[13] propuso una RNAA utilizada como un método de Análisis de Componentes Principales No-Lineales (ACPNL) para identificar y remover correlaciones entre las variables de un problema como una ayuda para reducir la dimensionalidad, visualización de datos, y análisis exploratorio de datos. ACPNL opera entrenando una red neuronal tipo feed-forward para llevar a cabo un mapeo idéntico de las entradas de la red y reproducirlas en la capa de salida. La red contiene una capa interna que actúa como un cuello de botella (la capa contiene una menor cantidad de neuronas que las capas de entrada o de salida), lo cual obliga a la red a desarrollar una representación compacta de los datos de entrada. La RNAA está compuesta por cinco capas. La Figura 1 muestra la arquitectura de la RNAA.

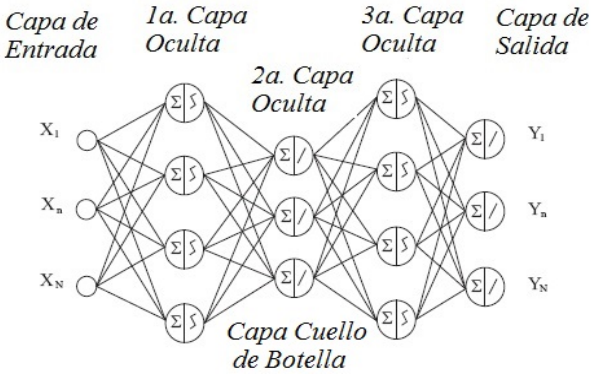


Fig. 1. Arquitectura de una RNAA

Esta RNAA tiene una capa de entrada y una capa de salida, cada una con N neuronas y tres capas ocultas con H_1, H_2, y H_3 neuronas respectivamente. Cuando una observación x es presentada en la entrada de la red, la ecuación de

la neurona de salida de la l^{th} capa es función de las neuronas en la $(l-1)^{th}$ capa, dada por la ecuación 1 y mostrada en la Figura 2.

$$y_j = f(\sum_{i=1}^{n(l-1)} w_{i,j}^{(l)} y_i^{l-1}) \;\; l = 1, ..., 4, \tag{1}$$

donde $y_j^{(0)} = x_n$, $n = 1, ..., N$ representa las componentes del vector de observaciones a la entrada de la red; $y_n^{(4)}$, $n = 1, ..., N$, representa las componentes de la estimación de x dada a la salida de la red; $n(l-1)$ da el número de neuronas en la $(l-1)^{th}$ capa. La función $f(.)$ es la función de activación de la neurona, la cual es sigmoidal. Las observaciones deben ser estandarizadas para que caigan dentro de un hipercubo unitario.

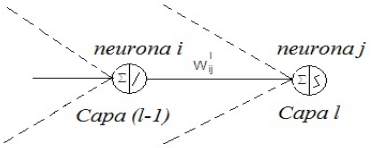


Fig. 2. Conexión de la i^{th} neurona de la $(l-1)$ capa con la j^{th} neurona de la l capa

Si la red tiene que aprender una tarea específica, es necesario ajustar los pesos de las conexiones entre las neuronas para minimizar la diferencia entre la salida esperada y la salida dada por la RNAA. Esta minimización es llevada a cabo cuando se calcula el error diferencial. El método más comunmente usado es el de retropropagación.

3.2. Máquina de soporte vectorial (MSV)

Sea $x_i (i = 1, ..., M)$ una entrada n-dimensional perteneciente a la Clase I o a la Clase II y las etiquetas asociadas $y_1 = 1$ para la Clase I y $y = -1$ para la Clase II. Para datos linealmente separables, puede determinarse un hiperplano $f(x)$ que separe los datos:

$$f(x) = w \cdot x + b = \sum_{j=1}^{n} w_j x_j + b. \tag{2}$$

Un hiperplano de separación satisface las restricciones que definen la separación de los datos muestrales, por ejemplo, $f(x_i) \geq +1$ si $y_i = +1$, y $f(x_i) \leq -1$ si $y_i = -1$. De esto resulta la ecuación 3:

$$y_i f(x_i) = y_i(w \cdot x_i + b) \geq 1, \;\; for \;\; i = 1, ..., M, \tag{3}$$

donde w es un vector n-dimensional y b es un escalar. La notación $w \cdot x_i$ corresponde al producto punto de los vectores w y x_i. El vector de pesos w

define la dirección del hiperplano de separación $f(x)$ y con w y b (bias) es posible definir la distancia del origen al hiperplano. El hiperplano de separación que tiene la máxima distancia entre el hiperplano y el dato más cercano es llamado hiperplano de separación óptimo. La Figura 3 muestra un ejemplo del hiperplano de separación óptimo de dos conjuntos de datos. Éste es perpendicular a la línea más corta entre las líneas de los límites de los dos conjuntos de datos y el plano, y la línea más corta que intersecta a cada uno a la mitad de la línea.

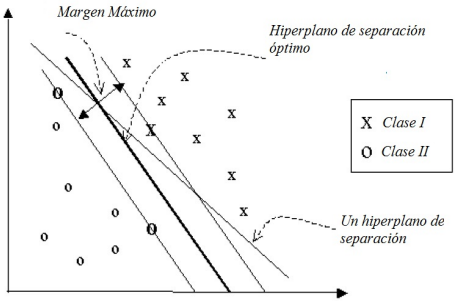


Fig. 3. Hiperplano de separación óptimo

El margen geométrico γ es la mitad de la suma de las distancias entre hiperplanos de separación arbitrarios y el dato negativo y positivo más próximo (x^- y x^+):

$$\gamma = \frac{1}{2}[(\frac{w}{\|w\|_2} \cdot x^+) - (\frac{w}{\|w\|_2} \cdot x^-)] = \frac{1}{2\|w\|_2}[(w \cdot x^+) - (w \cdot x^-)] = \frac{1}{\|w\|_2}. \quad (4)$$

Sin perder la generalidad el hiperplano de separación óptimo puede buscarse empleando las llamadas ecuaciones canónicas de hiperplanos, las cuales satisfacen $w \cdot x^+ + b = 1$ y $w \cdot x^- + b = -1$. El hiperplano óptimo maximiza el margen geométrico. Así el hiperplano óptimo puede encontrarse resolviendo el problema de optimización cuadrático presentado en la ecuación 5:

$$minimizar \ \frac{1}{2} \parallel w \parallel^2 \quad sujeto \ a \ \ y_i(w \cdot x_i + b) \geq 1. \quad (5)$$

La ecuación 5 puede ser simplificada considerablemente al convertir el problema con las condiciones de Karush-Kuhn-Tucker (KKT) al problema dual equivalente de Lagrange, obteniendo la función de decisión mostrada en la ecuación 6:

$$f(x) = \sum_{i=1}^{M} \alpha_i^* y_i x_i \cdot x + b^*. \quad (6)$$

Entonces el dato muestral desconocido x puede ser clasificado como Clase I si $f(x) > 0$ o como Clase II en caso contrario.

La MSV también puede aplicarse a casos donde los datos no sean linealmente separables. De esta manera la MSV mapea el vector de entrada en una dimensión espacial mayor. Este proceso esta basado en la selección de una función kernel. Algunas de las funciones kernel más empleadas son: lineales, polinomiales, de base radial y sigmoides. Adicionalmente entre los métodos mayormente aplicados para lograr una clasificación multiclase con la MSV está el conocido uno contra todos el cual construye M MSV donde M es igual al número de clases a ser clasificadas, ya que cada una de estas MSV separa una clase del resto. Otro método es conocido como uno contra uno $M*(M-1)/2$. Esta MSV combina su función de clasificación para determinar la clase a la cual la muestra de prueba pertenece mediante la acumulación de votos. La predicción con la mayoría de los votos proporciona la clasificación final.

4. Descripción de la propuesta

La presente propuesta es una metodología basada en el historial del proceso. Se necesitan bases de datos en modo de operación normal del sistema para entrenar una RNAA y llevar a cabo el proceso de detección. También son necesarias bases de datos que contengan información de las diferentes fallas que se pudieran presentar en el sistema que se monitorea. Con éstas bases de datos de las posibles fallas se da aprendizaje a la MSV multiclase, la cual es la encargada de dar el diagnóstico final cuando una fallas está presente. El proceso general de entrenamiento del sistema de detección y diagnóstico propuesto se muestra en la Figura 4.

El algoritmo para el proceso de entrenamiento se resume de la siguiente manera:

1. Localizar las mediciones provenientes de los sensores de las variables que se desean monitorear.
2. De manera aleatoria tomar un subconjunto ($\approx 80\,\%$) de la cantidad total de datos para entrenar las herramientas de softcomputing empleadas por la propuesta en la primera y segunda fase.
   Para la Priemera Fase
3. Tomar bases de datos de la operación normal del sistema.
4. Estandarizar el subconjunto de datos. Entrenar la RNAA y aprender el modelo.
5. Obtener los residuos de las condiciones de operación normal y sus correspondientes límites (umbrales).
   Para la Segunda Fase
6. Tomar bases de datos de las diferentes fallas que pudieran presentarse en el sistema.
7. Estandarizar el subconjunto de datos. Entrenar la MSV multiclase y aprender el modelo.
8. La MSV multiclase arroja el diagnóstico final.

Durante la primera fase del diagnóstico, el proceso de detección se lleva a cabo al obtener los residuos generados por la RNAA. Antes que todo, se debe

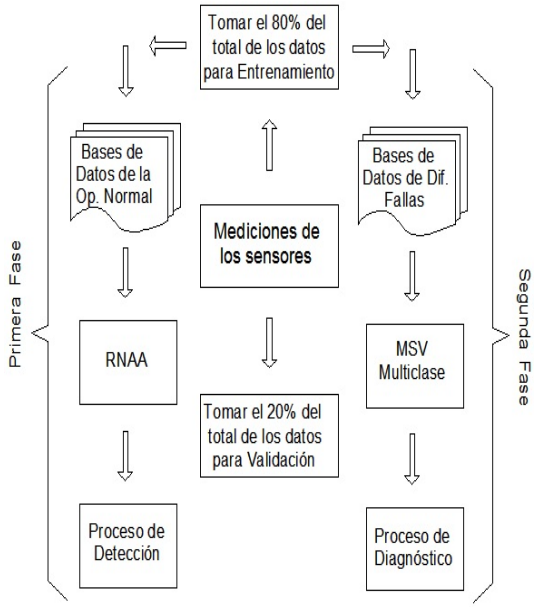


Fig. 4. Proceso general de entrenamiento del sistema de detección y diagnóstico propuesto

de realizar un proceso de estandarización para poder manipular las diferentes variables que se monitorean sobre una misma escala. Posteriormente se genera aleatoriamente un subconjunto de muestras formado por el 80 % del total de la base de datos. Dicho subconjunto de muestras es aprendido por la RNAA. Una RNAA de cinco capas es una red cuyas salidas son entrenadas para emular las entradas sobre un adecuado rango dinámico. Esta característica de la red es muy importante para monitorear variables de sistemas complejos que presentan un cierto grado de correlación entre ellas puesto que cada salida recibe información de cada una de las entradas. Durante el entrenamiento, para hacer que cada salida iguale a su correspondiente entrada, las interrelaciones entre todas las variables de entrada y cada salida individual se refleja en los pesos de conexión de la red. Como resultado de lo anterior se tiene que cada salida específica e incluso la correspondiente salida refleja solo una pequeña fracción del cambio de la entrada sobre un rango de valores razonablemente amplio. Ésto permite a la RNAA detectar la presencia de una falla al comparar simplemente cada una de las entradas con su correspondiente salida, obteniendo de esta manera los residuos. Posteriormente se calculan los límites de dichos residuos para condiciones de operación normal del sistema.

Una vez que el proceso de detección es llevado a cabo, la segunda fase de la propuesta empieza a trabajar. Ésta utiliza la MSV multiclase para clasificar la falla presente previamente detectada por la primera fase. La salida de la MSV multiclase indica cual variable se encuentra en modo de falla y el tiempo en

el cual dicha falla ha ocurrido, proporcionando de esta manera un diagnóstico completo del sistema que se monitorea. La Figura 5 muestra el esquema de operación en línea del sistema de diagnóstico propuesto.

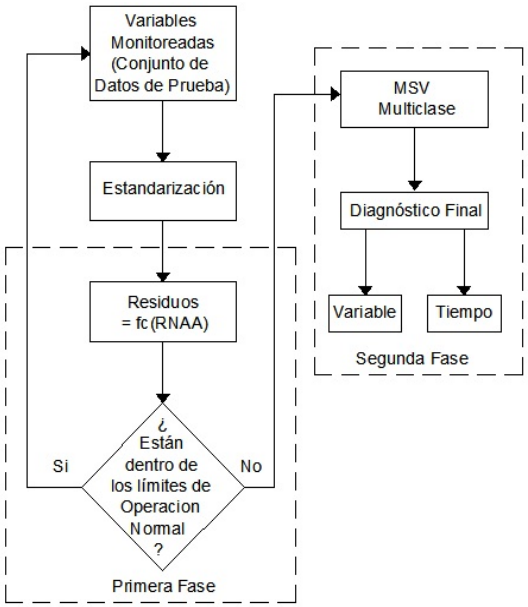


Fig. 5. Esquema de operación en línea del sistema de diagnóstico propuesto

5. Caso de estudio

La presente propuesta se validó y se puso en marcha en el Laboratorio de Manufactura de la Facultad de Ingeniería de la Universidad Autónoma de Coahuila. El cual cuenta con una celda de manufactura automatizada (CMA). Se hace mención que la CMA representa un sistema conformado por equipos de alta tecnología, en el que dichos equipos y componentes están integrados para realizar un proceso de manufactura completamente automatizado. Por lo anterior se tiene la necesidad de contar con un sistema de detección y diagnóstico de fallas que lleve a cabo un monitoreo de los equipos más susceptibles a la presencia de éstas y que intervienen dentro de un proceso en la CMA para evitar dejar fuera de operación a la misma. Es por lo anterior que la CMA representa un escenario complejo, perfecto para implementar el sistema de monitoreo, detección y diagnóstico de fallas en su banda transportadora de material.

5.1. Posibles fallas que se pueden presentar en el equipo monitoreado

La CMA cuenta con una banda transportadora para el traslado de pallets. Dichos pallets los toma un robot y los deposita en la banda transportadora. Una vez en la banda, ésta los transporta hacia su otro extremo en donde se emula un proceso de manufactura. Es por ello que se decide que el equipo a monitorear es la banda transportadora de material por representar un equipo que realiza una operación crítica dentro de la CMA así como dentro de un proceso de manufactura real. Debido a que el sistema de diagnóstico está basado en el historial del proceso, es que se puede monitorear todo tipo de variables ya que éstas son tratadas por la metodología solamente como un conjunto de números, sin importar si éstos son provenientes de un sensor de temperatura, presión, voltaje, frecuencia, corriente, etc. Lo cual dota de versatilidad a la metodología propuesta.

El sistema de la banda transportadora de material de la CMA está compuesto por 2 motores eléctricos de inducción trifásicos con rotor tipo jaula de ardilla.

Ya que dichos motores tienen una alimentación trifásica, se decide monitorear el voltaje de cada una de las líneas de alimentación para cada motor. Se considera que el problema de diagnóstico y detección de fallas de la banda transportadora representa una tarea retadora ya que como se comentó, las variables observadas serán los voltajes de alimentación de cada motor y como se pudo constatar en la revisión del estado del arte de sistemas eléctricos, se tiene que el monitoreo y diagnóstico de un sistema eléctrico no es una tarea trivial dada la presencia de características no lineales y presencia de ruido en la medición de las variables involucradas. En los motores eléctricos existe una gran cantidad de perturbaciones tanto internas como externas que afectan su funcionamiento originando de esta manera la presencia de fallas. Dentro del amplio espectro de condiciones que afectan a los motores eléctricos, el presente análisis se centra sobre las fallas que involucran el voltaje de alimentación de cada uno de los motores. Dichas fallas pueden ser fallas simétricas o fallas asimétricas.

1. Las fallas asimétricas se presentan cuando una, dos o las tres líneas de alimentación eléctrica quedan unidas al punto de referencia conocido como tierra.
2. Las fallas simétricas se presentan cuando dos de las léneas de alimentación eléctrica quedan unidas entre sí. Es decir, la magnitud de dos de las tres líneas disminuye y adicionalmente dichas líneas se colocan en fase.

El sistema de diagnóstico propuesto se adecuó al caso de estudio como se muestra en el esquema de la Figura 6 y se describe como sigue.

El desempeño de la metodología para el diagnóstico de múltiples fallas fue evaluado con 50 escenarios. Tales escenarios contenían fallas simétricas y asimétricas incluidas de manera aleatoria en los diferentes motores y sus respectivas líneas de alimentación eléctrica. Se realizó la adquisición y generación de bases de datos mediante la adquisición de ventanas de datos compuestas con 100 muestras por ser la forma sugerida de llevarlo a cabo de acuerdo con la revisión del estado del arte. Se consideraron tres posibles casos de información muestreada.

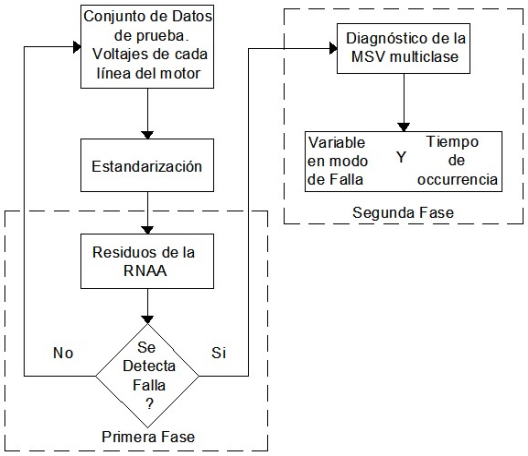


Fig. 6. Implementación de la metodología propuesta en el caso de estudio

- **Caso 1:** El sistema opera correctamente durante 25 muestras y en modo de falla durante 75 muestras.
- **Caso 2:** El sistema opera en modo normal durante 50 mestras y las 50 restantes en modo de falla.
- **Caso 3:** El sistema opera correctamente las primeras 75 muestras y las restantes 25 en modo de falla.

Falla simétrica. Se muestra como ejemplo la inducción de una falla simétrica entre las líneas LA y LC dentro de las primeras 25 muestras de la ventana del conjunto de datos de prueba. De acuerdo a la Figura 6 los datos de prueba se estandarizan y la primera fase de la metodología comienza a trabajar. La salida de la RNAA se muestra en la Figura 7 inciso (a). En ella se observa que la RNAA detecta la presencia de una falla puesto que existen datos que caen fuera del espacio de operación normal obtenido. Posteriormente comienza a trabajar la segunda fase de la metodología, en donde la MSV multiclase arroja el diagnóstico final indicando el tipo de falla presente y el tiempo de ocurrencia, lo cual se muestra en el inciso (b) de la Figura 7.

Falla asimétrica. También se realizó la inducción de una falla asimétrica, la cual fue líneas LA y LB a tierra a partir de la muestra 51 de la ventana del conjunto de datos de prueba. Los datos de prueba se estandarizan y la primera fase de la metodología comienza a trabajar. La salida de la RNAA se muestra en la Figura 8 inciso (a). En ella se observa que la RNAA detecta la presencia de una falla puesto que existen datos que caen fuera del espacio de operación normal obtenido. Posteriormente en la segunda fase de la metodología la MSV multiclase arroja el diagnóstico final indicando el tipo de falla presente y el tiempo de ocurrencia, lo cual se muestra en el inciso (b) de la Figura 8.

Las Tablas 1, 2 y 3 muestran el desempeño del sistema de diagnóstico de fallas para los tres casos respectivamente al considerar diferentes escenarios. Los

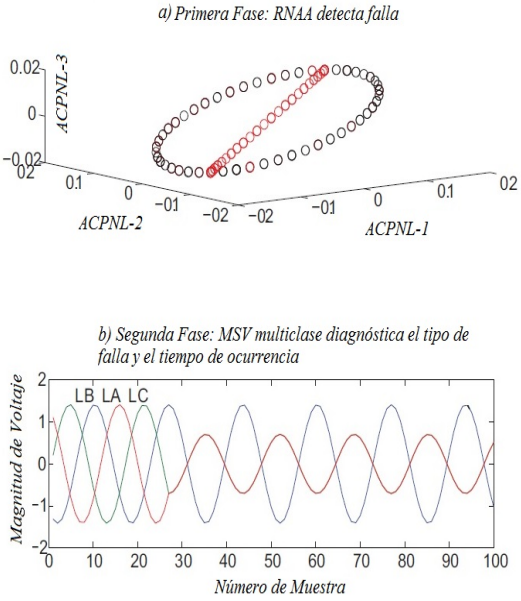


Fig. 7. Falla simétrica Caso 1

porcentajes mostrados corresponden a 50 diferentes escenarios para cada caso y la combinación de diferentes tipos de falla. La primer columna muestra el tipo de falla inducida en los motores. 1 es una línea a tierra, 2 son dos líneas a tierra, 3 son tres líneas a tierra, 4 y 5 representan falla entre dos líneas. La segunda columna da el porcentaje de detección de la falla. La tercera columna es el porcentaje de identificación de manera correcta del tipo de falla presente en el sistema. La cuarta columna muestra el porcentaje de la ubicación correcta del motor en modo de falla. La columna de precisión da el porcentaje de veces que el sistema realizó un diagnóstico correcto.

Tabla 1. Porcentajes de desempeño del sistema de diagnóstico para el Caso 1

Tipo de Falla Presente	Detección	Identificación	Localización	Precisión
1	100	100	100	100
2	100	100	100	100
3	100	100	100	100
4	90	90	90	90
5	90	90	90	90

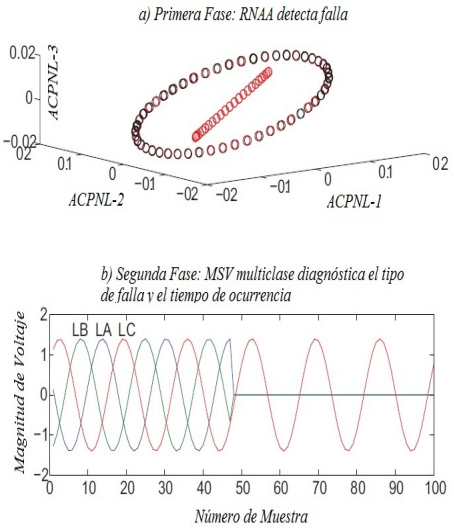


Fig. 8. Falla asimétrica Caso 2

Tabla 2. Porcentajes de desempeño del sistema de diagnóstico para el Caso 2

Tipo de Falla Presente	Detección	Identificación	Localización	Precisión
1	100	100	100	100
2	100	100	100	100
3	100	100	100	100
4	92	92	92	92
5	92	92	92	92

Tabla 3. Porcentajes de desempeño del sistema de diagnóstico para el Caso 3

Tipo de Falla Presente	Detección	Identificación	Localización	Precisión
1	100	100	100	100
2	100	100	100	100
3	100	100	100	100
4	95	92	92	93
5	95	92	92	93

6. Conclusiones

Se presentó una metodología para llevar a cabo un diagnóstico completo de sistemas complejos. La propuesta se basa en el análisis y manejo de los datos históricos del proceso. El diagnóstico final es llevado a cabo en dos fases. La

primera fase emplea una red neuronal autoasociativa para obtener los residuos entre el modo de operación normal previamente aprendido y la información de las mediciones provenientes de los sensores. Una vez que la primera fase detecta la presencia de una falla, la segunda fase comienza a trabajar para dar el diagnóstico final. La segunda fase utiliza una máquina de soporte vectorial multiclase, la cual clasifica la falla presente y proporciona su tiempo de ocurrencia.La metodología propuesta disminuye el problema de falsas alarmas debido a que toma en cuenta la correlación entre las variables monitoreadas mediante la RNAA.

Referencias

1. Z. Liao, F. Wen, W. Guo, X. He, W. Jiang, T. Dong, J. Liang, B. Xu: An Analytical Model and Otpimization Technique Based Methods For Fault Diagnosis in Power Systems. In: IEEE Third International Conference on Electric Utility Deregulation and Restructuring and Power Technologies, Nanjing, China, pp. 1388–1393 (2008)
2. R.A. Flauzino, V. Ziolkowski, I.N. da Silva, D.M.B. de Souza: Hybrid Intelligent Architecture for Fault Identification in Power Distribution Systems. In: PES'09, IEEE Power & Energy Society General Meeting, pp. 1–6 (2009)
3. F. Fahimi, D. Brown, M. Khalid: Feature Set Evaluation and Fusion for Motor Fault Diagnosis. In: IEEE Symposium on Industrial Electronics and Applications (ISIEA 2010), Penang, Malaysia, pp. 634–639 (2010)
4. I.Aydin. M. Karakose, E. Akin: FPGA Based Real Time Fuzzy Fault Detection Algorithm. In: IEEE International Conference on Soft Computing and Pattern Recognition (SoCPaR), pp. 389–394.
5. J.P. Nieto, L.E. Garza, R. Morales: Multiple Fault Diagnosis in Electrical Power Systems with Dynamic Load Changes Using Probabilistic Neural Networks. Computación y Sistemas, Vol. 14, No. 1, pp. 17–30 (2010)
6. K. Vinoth Kumar, S. Suresh Kumar, B. Praveena, J.P. John, J. Eldho Paul: Soft Computing Based Fault Diagnosis. In: IEEE Second International Conference on Computing, Communication and Networking Technologies, pp. 1–7 (2010)
7. L. Yuan, Y. He, J. Huang, Y. Sun: A New Neural Network Based Fault Diagnosis Approach for Analog Circuits by Using Kurtosis and Entropy as a Preprocessor. IEEE Transactions on Instrumentation and Measurement, Vol. 59, No. 3, pp. 586–595 (2010)
8. Q. Alsafasfeh, I. Abdel-Qader, A. Harb: Symmetrical Pattern and PCA Based Framework for Fault Detection and Classification in Power Systems. In: Conference on Electro Information Technology, pp. 1–5 (2010)
9. J.A. Jiang, C.L. Chuang, Y.C. Wang, C.H. Hung, J.Y. Wang, C.H. Lee, Y.T.Hsiao: A Hybrid Framework for Fault Detection, Classification, and Location Part I: Concept, Structure, and Methodology. IEEE Transactions on Power Delivery, Vol.26, No.3, pp. 1988–1998 (2011)
10. N. Qianwen, W. Youyuan: An Augmented Naive Bayesian Power Network Fault Diagnosis Method based on Data Mining. In: Asia-Pacific Power and Energy Engineering Conference (APEEC), pp. 1–4 (2011)
11. R.J. Romero, R. Saucedo, E. Cabal, A. Garca, R. A. Osornio, R.A. Salas, H. Miranda, N. Huber: FPGA-Based Online Detection of Multiple Combined Faults in Induction Motors Through Information Entropy and Fuzzy Inference. IEEE Transactions on Industrial Electronics, pp. 5623–5270, Vol. 58, No. 11 (2011)

12. J.P. Nieto: Multiple Fault Diagnosis in Electrical Power Systems with Dynamic Load Changes Using Soft Computing. In: 11th Mexican International Conference on Artificial Intelligence, MICAI 2012, San Luis Potosi, Mexico, Advances in Computational Science Proceedings, Part 2, pp. 319–330 (2012)
13. M. Kramer: Nonlinear Principal Component Analysis Using Autoassociative Neural Networks. AIChe Journal, Vol.37, No. 2, pp. 233–234 (1991)

Aprendizaje incremental basado en población como buena alternativa al uso de algoritmos genéticos

Luis A. Cruz Prospero, Manuel Mejía Lavalle, José Ruiz Ascencio, Virna V. Vela Rincón

Centro Nacional de Investigación y Desarrollo Tecnológico,
Departamento de Ciencias Computacionales,
Cuernavaca, Morelos, México

{lacp, mlavalle, josera, viryvela}@cenidet.edu.mx

Resumen. En la actualidad han surgido nuevos modelos computacionales que intentan superar a los modelos clásicos de optimización, este es el caso de la Computación Evolutiva, la cual se ha popularizado por los Algoritmos Genéticos y sus diferentes variantes que prometen ser mejores. En este artículo analizaremos las bondades y/o deficiencias del Algoritmo Genético básico y del algoritmo de Aprendizaje Incremental Basado en Población, el cual es un algoritmo de estimación de distribuciones que forma parte del paradigma de la Computación Evolutiva. Se presenta un estudio comparativo de ambos algoritmos que permite establecer, a partir de la experimentación realizada con 7 funciones objetivo, que el algoritmo de Aprendizaje Incremental Basado en Población presenta ventaja significativa en tiempo de ejecución de todas las pruebas, así como en la precisión obtenida en 6 de las 7 funciones objetivo analizadas. Aunque esta ventaja ya había sido reportada, en este artículo se ha experimentado con funciones multimodal con dos incógnitas y en tres dimensiones, que en la actualidad son consideradas difíciles de resolver.

Palabras clave: Optimización de funciones, computación evolutiva, algoritmos genéticos, algoritmo de estimación de distribuciones, aprendizaje incremental basado en población.

Population-Based Incremental Learning as Good Alternative for Genetic Algorithms

Abstract. At present, new computational models that attempt to overcome to the classical optimization models have emerged, this is the case of Evolutionary Computation, which has been popularized by Genetic Algorithms and their different variants that promise to be better. In this article we will discuss the benefits and/or shortcomings of basic Genetic Algorithm and Population-Based Incremental Learning algorithm, which is an estimation of distributions

algorithm and it is part of Evolutionary Computation´s paradigm. A comparative study of both algorithms is presented, here it is established from the experimentation with 7 objective functions that the Population-Based Incremental Learning algorithm presents significant advantage on runtime of all experiments as well as the accuracy obtained in 6 of 7 objective functions analyzed. Although this advantage had already been reported, in this paper we have experimented with multimodal functions with two variables and three dimensions which are considered difficult to solve nowadays.

Keywords: Optimization functions, evolutionary computation, genetic algorithms, estimation of distribution algorithm, population-based incremental learning.

1. Introducción

Los algoritmos de optimización han sido extensamente desarrollados por un largo tiempo. La optimización consiste en tratar variaciones, usando información, de un concepto inicial con el fin de mejorarlo. Muchos problemas de la industria ingenieril, en particular de los sistemas de manufactura, son de naturaleza muy compleja y difícil de resolver por técnicas convencionales de optimización [1].

La Computación Evolutiva se ha encargado de brindar algoritmos que den solución a problemas considerados complejos. Estos algoritmos basados en metaheurísticas cuentan con poblaciones que representan un conjunto de soluciones fiables de un problema dado y que con el uso de técnicas estocásticas hacen que estas poblaciones representen mejores soluciones a través del tiempo [2].

Uno de estos algoritmos son los Algoritmos Genéticos (AG), que desde su surgimiento en 1962, han sido populares dentro de la comunidad científica, siendo una técnica de búsqueda aleatoria y optimización perteneciente al campo de la Inteligencia Artificial que ha ganado gran aceptación. Su principal fundamento se basa en la Teoría de Evolución de Darwin y se complementa con otros conceptos y teorías más recientes del campo de la genética [3].

Los AG están basados en el modelo presentado por John Holland, a raíz de la publicación en 1975 de su libro "*Adaptation in Natural and Artificial systems*", consisten en métodos adaptativos, usados en problemas de búsqueda y optimización de parámetros, basados en la reproducción sexual y el principio de supervivencia del más apto [4].

Una definición más completa es dada por Goldberg: "Los Algoritmos Genéticos son algoritmos de búsqueda basados en la mecánica de la selección natural y de la genética natural. Combinan la supervivencia del más apto entre estructuras de secuencias con un intercambio de información estructurado, aunque aleatorizado, para constituir así un algoritmo de búsqueda que tenga algo de las genialidades de las búsquedas humanas" [5].

Recientemente aparecieron los Algoritmos de Estimación de Distribuciones (EDA), una nueva familia de algoritmos evolutivos, desarrollados con el objetivo principal de

evitar la interrupción de soluciones parciales, teniendo como principal diferencia con los AG, que no existen las operaciones de cruce ni de mutación.

Dentro de la clasificación de los EDA se encuentra el algoritmo de Aprendizaje Incremental Basado en Población (PBIL por sus del inglés, *Population-Based Incremental Learning*), un algoritmo simple que utiliza un vector de probabilidad para generar la población, tomando en cuenta las evaluaciones más altas del vector [6].

Desde que PBIL fue propuesto por primera vez ha demostrado tener superioridad ante los AG [7], sin embargo en este trabajo se presenta el uso de PBIL en distintas pruebas realizadas en la optimización de funciones más complejas, como lo son las funciones multimodal con dos incógnitas.

Este artículo tiene como objetivo comparar el desempeño del AG y PBIL en un espacio de búsqueda amplio, con el fin de determinar la superioridad de uno de ellos en la maximización de funciones objetivo con alto grado de dificultad.

En la Sección 2 se brinda una descripción del problema que se pretende resolver a través de la Computación Evolutiva, así como un breve análisis del AG y PBIL. En la Sección 3 se describen cada una de las funciones objetivo que servirán para el análisis de los algoritmos, mientras que en la Sección 4 se discuten los resultados obtenidos en las pruebas. Finalmente en la Sección 5 se exponen las conclusiones.

2. Optimización por medio de computación evolutiva

2.1. Optimización de funciones

Los sistemas artificiales usados en ingeniería, a menudo son modelados a través de modelos matemáticos que cuentan con funciones y restricciones que definen ciertos procesos. Dichos modelos pueden contar con funciones de tipo lineales o no lineales, las cuales juegan un papel primordial en la descripción cuantitativa y el análisis de problemáticas inmersas en distintas áreas de la industria. Muchos problemas prácticos del mundo real, como la optimización combinacional, diseños de módulos y planeación de rutas, pueden ser transformados en funciones multimodal a través de del modelado y la simulación [8].

Generalmente es difícil resolver problemas sin una formulación matemática; un modelo matemático puede ser usado para representar problemas del mundo real con menos dificultad, participando en la optimización de parámetros que den solución a dichos problemas de forma satisfactoria.

La optimización de funciones es el proceso de encontrar un conjunto de posibles puntos dentro de una región factible de cierto espacio de búsqueda. Lo anterior es con el objetivo de que el valor resultante de la función se convierta, ya sea en el máximo o mínimo, en función del problema y los requerimientos.

Básicamente los problemas de optimización de funciones están compuestos por las siguientes tres partes [9]:

- Función objetivo. Es el modelo matemático del problema de la vida real a resolver.

- Conjunto de variables. Estas variables afectan directamente el valor de la función objetivo.
- Conjunto de restricciones. Las variables pueden tomar ciertos valores y no pueden tomar otros, estos dependerá de las restricciones definidas en el problema.

2.2. Algoritmos basados en computación evolutiva

Algoritmo Genético. Los AG son métodos que proveen de flexibilidad para la búsqueda efectiva de máximos o mínimos globales en problemas de optimización; esto quiere decir que se encargan de encontrar el mejor resultado posible dentro de un conjunto admisible de condiciones para lograr cierto objetivo. Estos algoritmos trabajan con aleatoriedad, usando la inteligencia a la manera de la Madre Naturaleza, probando eventos a lo largo de los años y la supervivencia del más apto.

Estos algoritmos trabajan sobre una población de individuos, donde cada uno de ellos representa una posible solución, cada individuo está compuesto de características, llamadas genes, estas soluciones son codificadas, generalmente por una cadena binaria. La longitud de la cadena dependerá del dominio de los parámetros, así como de la precisión requerida y es definida en (1).

$$2^{m_j-1} < \left(b_j - a_j\right) * 10^n \leq 2^{m_j} - 1 \tag{1}$$

En donde $[a_j, b_j]$ es el dominio establecido para la búsqueda; n es el número de dígitos de precisión requeridos después del punto decimal; mientras m_j serán los bits demandados por el problema.

Como en la naturaleza, la selección proporciona el mecanismo que conduce a mejores soluciones para sobrevivir, estas son asociadas a un valor de aptitud para reflejar qué tan buena es comparada con otras soluciones.

El valor de aptitud está arraigado a la función objetivo en la que se pretende realizar una búsqueda u optimización. Para realizar dicha evaluación es necesario convertir las cadenas binarias a números reales y esto se logra a partir de los siguientes pasos [1]:

- Convertir la cadena binaria a base 10 (Z):

$$z = \sum_{i=0}^{n} b_i 2^i \tag{2}$$

En donde n es la longitud de la cadena y b es valor del bit (0 o 1).

- Calcular el correspondiente número real x a través de:

$$x = a_j + z\left(\frac{b_j - a_j}{2^{m_j} - 1}\right) \tag{3}$$

Los valores más aptos son los que tiene mayores posibilidades de reproducirse en la siguiente generación. La cruza es la combinación del material genético entre los padres, intercambiando partes de las cadenas para producir un nuevo individuo. Por

último, la mutación causa cambios aleatorios en la cadena alterando los genes con una probabilidad establecida. Este ciclo es repetido hasta que se alcanza un criterio de paro, por ejemplo, un número definido de generaciones o cuando no haya cambios en la población [10].

Sintetizando, los pasos a seguir para implementar un AG son los siguientes:

- Generar una población aleatoriamente.
- Evaluar la aptitud de cada uno de los individuos y seleccionar los más aptos.
- Cruzar los individuos para generar una nueva población.
- Mutar aleatoriamente algunos individuos de la población.
- Repetir desde el paso dos hasta cumplir un criterio de paro.

Aprendizaje Incremental Basado en Población. Por otra parte tenemos a PBIL, este algoritmo es el más popular dentro de la clasificación EDA. El algoritmo estándar es una combinación de optimización evolutiva y aprendizaje competitivo [11]. Este algoritmo trabaja con un vector de probabilidades (V_P), el cual codifica una distribución de probabilidad que representa un prototipo de buenas soluciones y que se usa para generar una población de posibles soluciones (vectores binarios) en cada iteración [12]. Cada valor dentro del vector representa la probabilidad de que se pueda generar un uno o un cero, en la posición del gen correspondiente, inicialmente todos los valores tienen el 0.5 de probabilidad.

En cada generación, usando el V_P se obtienen los individuos, cada uno de éstos es evaluado y sólo los mejores son seleccionados. Los individuos seleccionados son usados para actualizar el V_P. La ejecución del algoritmo se detiene cuando el V_P converge, es decir, cuando todos sus elementos sean cero o uno [13].

La actualización del V_P está dado por (4).

$$V_P = V_P * (1.0 - LR) + Mejor\ individuo * L_R, \tag{4}$$

en donde:

- $V_P = Vector\ de\ probabilidad$
- $L_R = Taza\ de\ aprendizaje$

A continuación, se resumen los pasos del algoritmo PBIL para generar una solución:

- Inicializar todos los valores del V_P a 0.5.
- Generar un conjunto de individuos mediante el V_P.
- Evaluar la aptitud de cada uno de los individuos y ordenarlos de acuerdo a su aptitud.
- Actualizar el V_P de acuerdo al mejor individuo.
- El programa se repite desde el paso 2 y finaliza de acuerdo al criterio establecido.

Para actualizar el V_P se debe tomar en cuenta la Tasa de Aprendizaje (L_R), es de fundamental importancia en la implementación, ya que determina la rapidez y exactitud para la obtención de los resultados finales. En otras palabras, el L_R, es el factor de importancia que se le da al mejor individuo para la actualización del V_P.

Existen variantes para mejorar el rendimiento de este algoritmo, pero para este trabajo consideramos el uso del PBIL original.

3. Funciones implementadas y medidas de desempeño

Los problemas reales de optimización, ya sean de minimización, o maximización como es el caso que se abordará en este trabajo, comúnmente cuentan con la característica de no tener una única solución que satisfaga el problema [14]. Es por eso que se aborda, en el desarrollo de este trabajo, un conjunto de siete funciones multimodal [15], con el fin de localizar el máximo global que satisfaga a cada una de ellas por medio de la Computación Evolutiva.

- *Parabolic*. La función *Parabolic* resulta en una gráfica en forma de curva, la cual contiene un vértice y un eje de simetría que divide por la mitad, en forma vertical, a dicha curva dando como resultado dos mitades que son imágenes espejo, una de la otra. La representación en tres dimensiones de esta función se muestra en Fig. 1(a) y es definida por (5):

$$f1(x,y) = x^2 + y^2. \tag{5}$$

- Peaks. Es una función con dos variables. Obtenida por la traslación y ampliación de las distribuciones Gaussianas [16]. Tiene múltiples picos y valles, como se observa en Fig. 1(b). Esta función está definida por (6).

$$\begin{aligned} f2(x,y) = {} & 3(1-x)^2 e^{-(x^2+(y+1)^2)} - 10\left(\frac{x}{5} - x^3 - y^5\right) e^{-(x^2+y^2)} \\ & - \frac{1}{3} e^{-((x+1)^2+y^2)}. \end{aligned} \tag{6}$$

- *Himmelblaus*. La función modificada de *Himmelblaus* cuenta con 4 puntos que maximizan la función objetivo. Dicha función está definida en (7) y se muestra gráficamente en la Fig. 1(c).

$$f3(x,y) = 200 - (x^2 + y^2 - 11)^2 - (x + y^2 - 7)^2. \tag{7}$$

- *Equal peaks*. Como se muestra en la Fig. 1(d) la función *Equal peaks* está constituida por diferentes picos que tienen los mismos máximos [17], esta función se define por (8).

$$f4(x,y) = \cos(x)^2 + sen(y)^2. \tag{8}$$

- *Rastrigin*. Esta función fue propuesta por Rastrigin como una función bidimensional [18]. Esta función presentan una alta dificultad debido al amplio espacio de búsqueda y al alto número de máximos y mínimos locales. Mientras que función *Peaks* consiste en únicamente 3 picos, la función *Rastrigin* consiste en alrededor de 100 picos en un rango considerado [19], como se muestra en la Fig. 1(e). La función *Rastrigin* está definida por (9).

$$f5(x,y) = 20 + (x^2 - 10\cos(2\pi x) + y^2 - 10\cos(2\pi y)). \quad (9)$$

- *Circles*. La función *Circles* contiene múltiples círculos concéntricos (ver Fig. 1(f)). A diferencia de las funciones *Peaks* y *Rastrigin* en donde cada pico es un solo punto, la función *Circles* presenta líneas circulares de picos locales que contienen infinitos picos [20] y son definidos por (10).

$$f6(x,y) = (x^2 + y^2)^{0.25}((sen(50(x^2 + y^2)^{0.1}))^2 + 1.0). \quad (10)$$

- *Schaffer*. Esta función es considerada para las pruebas realizadas en este trabajo debido a la complejidad presentada al contar con diversos máximos y mínimos locales. Se muestra gráficamente en la Fig. 1(g) y es definida por (11).

$$f7(x,y) = 0.5 + \frac{(sen(\sqrt{x^2 + y^2}))^2 - 0.5}{(1 + 0.1(x^2+y^2))^2}. \quad (11)$$

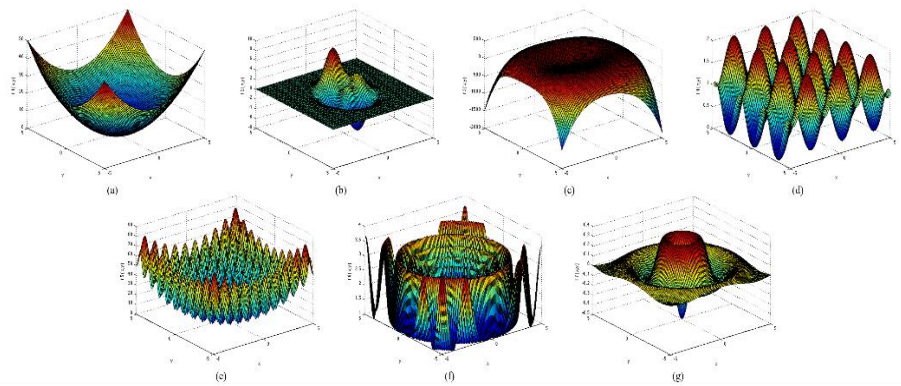


Fig. 1. Funciones objetivo

Con el fin de evaluar el desempeño de los algoritmos AG y PBIL se realizó la búsqueda de los valores que maximizan cada una de las funciones objetivo descritas anteriormente. En la Tabla 1 se muestra el óptimo global que maximiza cada función.

Estos datos servirán como base para la evaluación de cada algoritmo; la cual se realizará a través de cálculo de la precisión del resultado, así como el tiempo de ejecución.

4. Resultados y discusión

Para realizar la experimentación se implementó una plataforma propia de experimentación desarrollada en lenguaje Java, la cual facilitó la modificación de cada uno de los parámetros de ambos algoritmos. Este mismo *software* permitió observar el comportamiento de cada uno de los algoritmos en el proceso de maximización de las siete funciones objetivo establecidas anteriormente.

Con el objetivo de comprobar la eficacia de los AG y PBIL en la búsqueda de maximizar las funciones multimodal, se estableció un espacio de búsqueda en un rango de [-100,100]. Además los parámetros *x* e *y* de cada función se codificaron con cadenas binarias de 15 bits cada una, dando como resultado individuos con 30 genes.

Se plantearon los siguientes parámetros iniciales, para la realización de diferentes pruebas con cada una de las funciones objetivo:

- Número de individuos, 20, 100, 300 y 500.
- Número de generaciones, 20, 200 y 500.

Tabla 1. Óptimos globales

Función objetivo	Óptimo Global
f1(x,y)	20000
f2(x,y)	8.116
f3(x,y)	200
f4(x,y)	2
f5(x,y)	20000
f6(x,y)	23.309
f7(x,y)	0.831

Cada algoritmo cuenta con parámetros que se pueden someter a cambios para, de tal modo, obtener distintos comportamientos [2,7], por lo cual se aplicarán cada una de las pruebas con variaciones en éstos, en el caso de AG su variante se encuentra en la Probabilidad de Mutación (P_M) y Probabilidad de Cruce (P_C) y en PBIL se trabajará variando L_R.

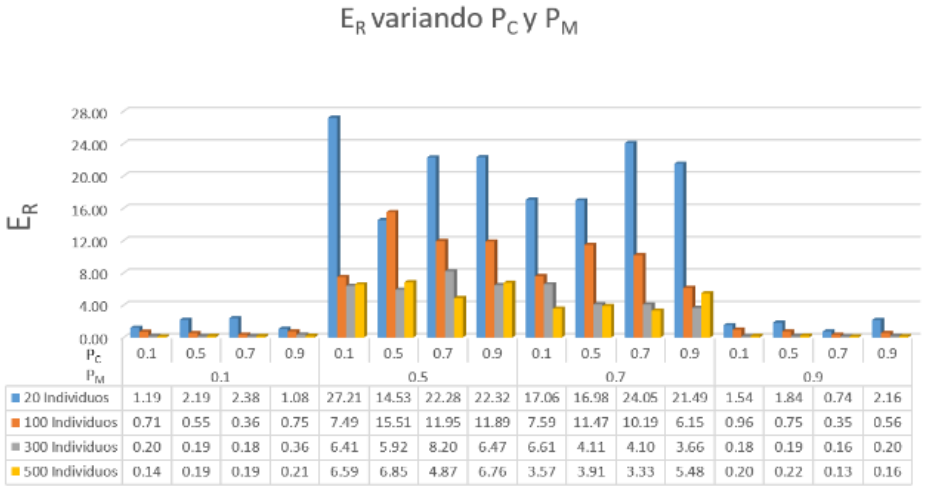
Para la evaluación de cada experimento se ejecutó el programa con los mismos parámetros, excepto la semilla, de forma independiente 5 veces y se reporta el promedio de ésto.

Con el fin de realizar un análisis más detallado de cada algoritmo, primeramente se mostrarán los resultados obtenidos de forma individual. En el caso del AG se experimentó variando la P_M y la P_C. En la Fig. 2 se ejemplifica, usando la función objetivo $f5(x,y)$, el índice de Error Relativo (E_R) obtenido por AG con cada combinación de ambos parámetros, adicionalmente se muestra la variación obtenida con diferente número de individuos.

De acuerdo a los resultados obtenidos en los experimentos, mostrados en la Fig. 2 se logra destacar que el AG presenta dificultades en la precisión de sus resultados cuando la P_M es igual o superior a 0.5, sin embargo al tener una P_M alta, cercana a 1, el E_R vuelve a disminuir; pero cabe destacar que al realizar esta acción se estaría perdiendo el esquema genético característico de los AG, debido a que una P_M cercana a 1 haría mutar a todos los individuos.

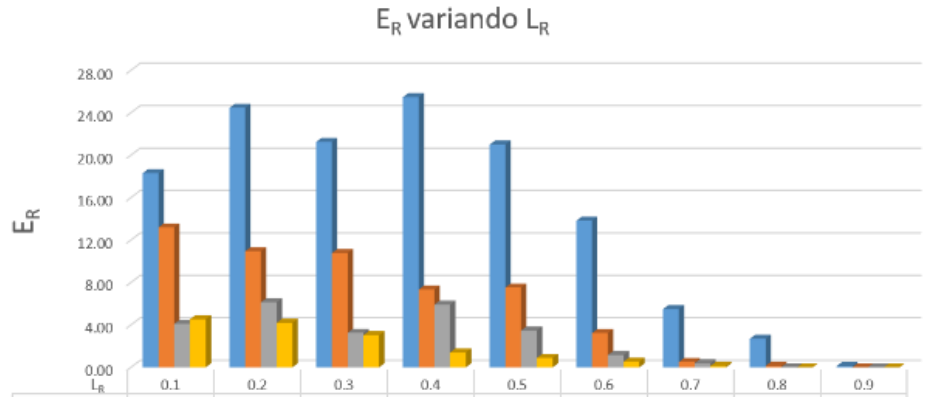
Por otro lado, se observa que al tener una P_C alta (más cercana a 1) el E_R disminuye, aumentando las posibilidades de éxito del algoritmo para localizar el óptimo global.

Pasando a PBIL, se usó como variable en los experimentos el L_R, se midió el desempeño del algoritmo calculando el E_R con diferente número de individuos. Los resultados obtenidos demuestran que al tener un L_R cercano a 1, los resultados tienden a mejorar rápidamente en cada iteración, como se muestra en la gráfica de la Fig. 3.



P_M	0.1				0.5				0.7				0.9			
P_C	0.1	0.5	0.7	0.9	0.1	0.5	0.7	0.9	0.1	0.5	0.7	0.9	0.1	0.5	0.7	0.9
20 Individuos	1.19	2.19	2.38	1.08	27.21	14.53	22.28	22.32	17.06	16.98	24.05	21.49	1.54	1.84	0.74	2.16
100 Individuos	0.71	0.55	0.36	0.75	7.49	15.51	11.95	11.89	7.59	11.47	10.19	6.15	0.96	0.75	0.35	0.56
300 Individuos	0.20	0.19	0.18	0.36	6.41	5.92	8.20	6.47	6.61	4.11	4.10	3.66	0.18	0.19	0.16	0.20
500 Individuos	0.14	0.19	0.19	0.21	6.59	6.85	4.87	6.76	3.57	3.91	3.33	5.48	0.20	0.22	0.13	0.16

Fig. 2. Error relativo obtenido con distintas P_C y P_M



L_R	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
20 Individuos	18.30	24.46	21.25	25.47	21.00	13.82	5.50	2.69	0.11
100 Individuos	13.20	10.94	10.77	7.33	7.52	3.23	0.50	0.12	0.00
300 Individuos	4.10	6.13	3.25	5.91	3.47	1.15	0.39	0.02	0.00
500 Individuos	4.53	4.20	3.03	1.41	0.89	0.54	0.13	0.00	0.00

Fig. 3. Error relativo obtenido con distintos L_R

De acuerdo a los resultados obtenidos en este trabajo, se externa una característica fundamental que hace que PBIL sobresalga a AG más allá de la precisión, y es el tiempo. Esto es debido a que, como se muestra en la Fig. 4, el tiempo de ejecución del algoritmo PBIL no tiende a tener un alto crecimiento al aumentar el número de individuos de la población, mientras tanto en el AG existe un aumento considerable,

como ejemplo, se puede observar que el tiempo de ejecución de PBIL con 500 individuos en la población es de 53.4 ms, mientras que AG le tomó un tiempo de 24,227 ms; tomándole al AG un tiempo que es 453 veces más que el tiempo empleado por PBIL.

Fig. 4. Tiempo de ejecución de AG vs PBIL

Otra característica interesante en el análisis de ambos algoritmos es la forma en que van evolucionando generación a generación debido a que esto es determinante al momento de obtener mejores resultados en menos generaciones. En la Fig. 5, con ayuda de la plataforma de experimentación, podemos observar como PBIL presenta una rápida estabilidad desde la generación 8, mientras que el AG, debido a su aleatoriedad en cada una de sus etapas presenta una alta variabilidad.

Con el fin de brindar una comparativa más amplia de ambos algoritmos en el proceso de maximización de las 7 funciones propuestas, en la Fig. 6 se plasma una gráfica que evidencia la precisión de AG y PBIL.

Para la gráfica de la Fig. 6 se usa como parámetros iniciales, en el caso de AG: P_C=0.9 y P_M=0.1; y en PBIL: L_R=0.9. Estos valores fueron asignados debido a que en general fueron los que obtuvieron mejores resultados en las pruebas.

Como se observa en la Fig. 6 el algoritmo PBIL presenta mejores resultados en la precisión de los funciones objetivo: $f1(x,y)$, $f4(x,y)$, $f5(x,y)$, $f6(x,y)$ y $f\,7(x,y)$. En estas funciones PBIL combina eficiencia con eficacia, sin embargo en $f2(x,y)$ y $f3(x,y)$ el AG resulta ser más preciso.

Finalmente en la Tabla 2 se muestran los mejores resultados obtenidos para cada función objetivo, así como los parámetros usados por cada algoritmo para alcanzar estos resultados.

Con la información de la Tabla 2 se reafirma la superioridad de PBIL en la optimización de funciones debido a su mayor precisión y sobre todo, el tiempo de ejecución con el que logra brindar resultados favorables, debido a que el tiempo promedio de ejecución de los resultados observados en la Tabla 2 es de 178,067.257 ms para AG, mientras que PBIL le tomó un tiempo promedio de 433.25 ms, obteniendo PBIL resultados 411 veces más rápido que el AG.

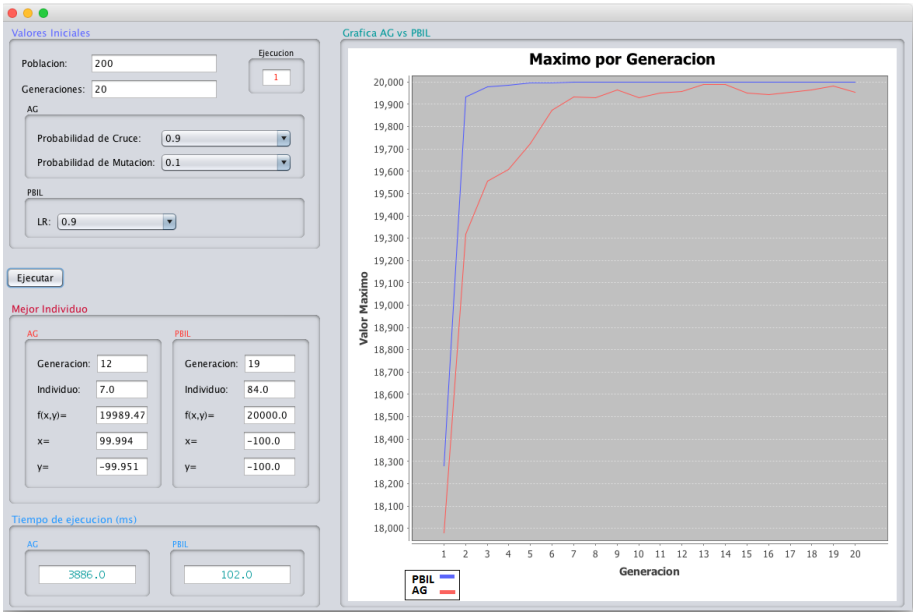


Fig. 5. Estabilidad de AG y PBIL

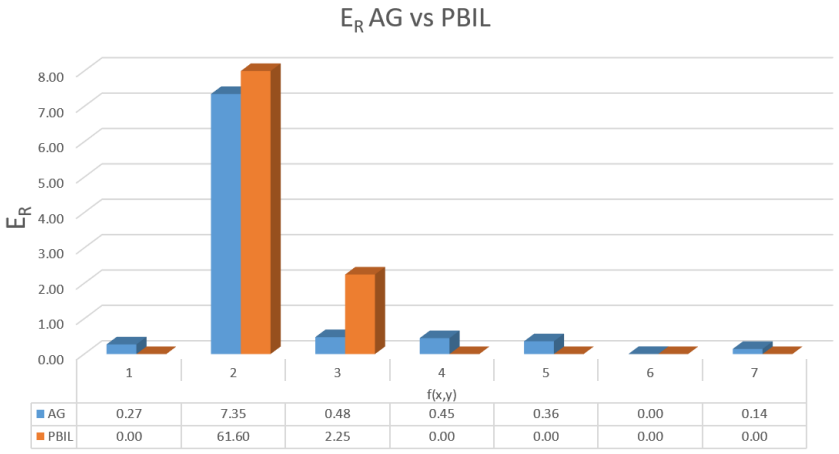


Fig. 6. *E*$_R$ por función (AG vs PBIL)

En precisión PBIL logra alcanzar el 100% en 5 de las 7 funciones evaluadas. Mientras que el AG destaca en f2(x,y) con un 26.28% de superioridad, esto es debido a que como se observa en la Fig. 2(b) existen máximos locales que se encuentran distantes de los demás puntos, con lo que debido a su aleatoriedad en la mutación el AG logra superar. A su vez en la Tabla 2 se aprecia que para AG es requerido un mayor número de individuos, así como de generaciones para alcanzar resultados más cercanos al óptimo, siendo una desventaja apreciable que repercute en el tiempo de ejecución.

Tabla 2. Mejores resultado AG y PBIL

Función	*f1*	*f2*	*f3*	*f4*	*f5*	*f6*	*f7*
			AG				
Generaciones	20	20	500	200	500	20	20
Individuos	500	500	500	500	500	300	500
P_C	0.50	0.90	0.90	0.50	0.50	0.50	0.90
P_M	0.10	0.10	0.10	0.10	0.90	0.10	0.90
E_R	0.17	0.66	0.06	0.02	0.11	0.00	0.00
Tiempo	23829.20	23912.00	466787.80	201536.20	502031.40	7287.60	21087.40
Precisión	99.83	99.34	99.94	99.98	99.89	100.00	100.00
			PBIL				
Generaciones	20	500	500	20	20	20	20
Individuos	100	500	500	500	100	100	300
L_R	0.90	0.70	0.80	0.90	0.90	0.80	0.90
E_R	0.00	26.94	0.64	0.00	0.00	0.00	0.00
Tiempo	11.40	1597.20	1325.00	38.40	12.00	12.80	36.00
Precisión	100.00	73.06	99.36	100.00	100.00	100.00	100.00

5. Conclusiones

La Computación Evolutiva en general tiene gran presencia en la actualidad debido a las aportaciones que ésta ha tenido en la resolución de problemas de optimización.

Desde un AG clásico hasta las diferentes variantes que existen actualmente, como lo es PBIL son herramientas poderosas, siempre y cuando se opte por el algoritmo ideal para cada problema.

En referencia a los resultados obtenidos se puede observar que en el caso de estudio de este trabajo, el cual fue la maximización de funciones multimodal, el algoritmo PBIL presenta una ventaja significativa en cada uno de los rubros evaluados en la fase de experimentación y presentados en la fase de resultados.

Se observa que PBIL presenta resultados más cercanos al óptimo en menor tiempo posible, es por eso que se puede concluir en este artículo que es una mejor opción en la optimización de funciones de maximización, sin embargo PBIL quizás podría ser mejorado si se incluye el mecanismo de mutación en los genes de los individuos: esto será motivo de investigación de nuestro trabajo futuro.

Aunque la Computación Evolutiva trabaja con datos aleatorios, PBIL presenta una ventaja al sustentarse en la probabilidad para garantizar más certeza en sus resultados.

Finalmente de acuerdo a los resultados de la experimentación se observó que para obtener mejores resultados en ambos algoritmos es recomendable usar una población

del doble del número de genes de los individuos. Además, al realizar los experimentos con PBIL se notó que un L_R de 0.9 funciona bien la mayoría de las veces, por lo que sugerimos probar de inicio con este valor.

Referencias

1. Guo, P. Wang, X. Han, Y.: The Enhanced Genetic Algorithms for the Optimization Design. In: 3rd International Conference on Biomedical Engineering and Informatics (BMEI 2010)
2. Khaparde, A.R., Raghuwanshi, M.M., Malik, L.G.: A New Distributed Differential Evolution Algorithm. In: International Conference on Computing, Communication and Automation (ICCCA) (2015)
3. Monzón, C.F., Mariño, R., Bogado, S.L. Validación de la implementación de la Función de Aptitud de un Algoritmo Genético aplicado a la sectorización. Universidad Nac. Del Nord. Comun. Científicas y Tecnológicas, pp. 7–10 (2005)
4. Withley, D.: A genetic algorithm tutorial. Stat. Comput., Vol. 4, No. 2, pp. 65–85 (1994)
5. Gestal, M.: Introducción a los algoritmos genéticos. (2010)
6. Rastegar, R., Hariri, A.: The Population-Based Incremental Learning Algorithm converges to local optima. Nuerocomputing, Vol. 69, No. 13-15, pp. 1772–1775 (2006)
7. Baluja, S.: Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning. Carnegie Mellon University, Pittsburgh, PA, USA, Tech. Rep. CMU-CS-94-163 (1994)
8. Liu, X., Yang, Z., Pan, W.: An improved adaptive immune genetic algorithm based on information entropy. In: International Conference on Industrial Informatics-Computing Technology, Intelligent Technology, Industrial Information Integration (2015)
9. Patalia, T.P., Kulkarni, G.R.: Behavioral Analysis of Genetic Algorithm for Function Optimization. C. U. Shah College of Engineering & Technology, Surendranagar, India, (2010)
10. Rodríguez, R.P., Aguirre, A.H.: Un algoritmo de Estimación de Distribuciones para el problema de secuenciamineto en configuración jobshop flexible. (2005)
11. Mavrovouniotis, M., Yang, S.: Population-Based Incremental Learning with Immigrants Schemes in Changing Environments. IEEE Symposium Series on Computational Intelligence (2015)
12. Fernandez de Viana, I., Cordón, O., Alonso, S., Herrera, F.: La metahurística de optimización basada en colonias de hormigas: modelos y nuevos enfoques. Optim. Intel., pp. 261–314 (2004)
13. Gosling, T., Tsang, E.: Population Based Incremental Learning with Guided Mutation Versus Genetic Algorithms: Iterated Prisioners Dilemma. (2005)
14. Lee, J., Song, S., Yang, Y., Shim, H., Lee, H., Lee, K., Yoon, Y.: Multimodal Function Optimization Based on the Survival of the Fitness kind of the Evolution Strategy. Proceedings of the 29th Annual International Conference of the IEEE EMBS Cité Internationale, Lyon, France, August 23-26 (2007)
15. Hua, Q., Wu, B.: Peak Detection Method for Multimodal Function Optimization. Proceedings of the Sixth International Conference on Machine Learning and Cybernetics, Hong Kong, 19-22 August (2007)
16. Reutskiy, S.Y., Chen, C.S.: Approximation of multivariate functions and evaluation of particular solutions using Chebyshev polynomial and trigonometric basis functions. International Journal for Numerical Methods in Engineering, Vol. 67, No. 13, pp. 1811–1829 (2006)

17. Parsopoulos, K., Vrahatis, M.N.: On the computation of all global minimizers through particle swarm optimization. IEEE Transactions on Evolutionary Computation, Vol. 8, No. 3, pp. 211–224 (2004)
18. Törn, A., Zilinskas, A.: Global optimization. New York: Springer (1989)
19. Krishnanand, K.N., Ghose, D.: Glowworm swarm optimization for simultaneous capture of multiple local optima of multimodal functions. Swarm Intell, Vol. 3, pp. 87–124 (2008)
20. Muller, S.D., Marchetto, J., Koumoutsakos, S.A.P.: Optimization based on bacterial chemotaxis. IEEE Transactions on Evolutionary Computation, Vol. 6, No. 6, pp. 16–29 (2002)

Diseño de un procesador analógico-digital para la implementación integrada de redes neuronales en sistemas de bajo consumo

Javier Alejandro Martínez Nieto[1], María Teresa Sanz Pascual[1], Nicolás Medrano Marqués[2], Belén Calvo López[2]

[1] Instituto Nacional de Astrofísica, Óptica y Electrónica (INAOE), Departamento de Electrónica, Puebla, México

[2] Universidad de Zaragoza, Grupo de Diseño Electrónico (GDE), Zaragoza, España

{almartinez,materesa}@inaoep.mx, {nmedrano,becalvo}@unizar.es

Resumen. En este artículo se presenta el diseño electrónico en tecnología CMOS estándar de 0.18μm con alimentación $V_{DD} = 1{,}8V$, de los principales bloques que conforman una neurona: el multiplicador y la función de activación no-lineal. De igual forma, se presentan los resultados por simulación eléctrica en CADENCE, así como el modelado matemático en MATLAB de su comportamiento. Una comparación de ambos modelos presenta errores relativos $e_r < 1\,\%$ para las dos operaciones. Para su validación, los modelos matemáticos generados fueron aplicados a una estructura de red neuronal entrenada para resolver la operación lógica XOR.

Palabras clave: Red neuronal artificial, CMOS, calibración de sensores.

Design of an Analog-digital Processor for Integrated Implementation of Neural Networks in Low-power Systems

Abstract. The electronic design of the artificial neuron main characteristic blocks is presented in this paper. Both, the multiplier and the non-linear activation function, were designed in standard $0{,}18\mu m$ CMOS process with $1{,}8V$ power supply. CADENCE electrical simulation results and mathematical modeling in MATLAB are also presented. A comparison between the high level and electrical models shows a relative error below 1 % for both operations. In order to verify the correct operation, the generated models were applied to a trained neural network structure to solve the XOR logical operation.

Keywords: Artificial neural networks, CMOS, sensor calibration.

1. Introducción

Los sistemas de inteligencia artificial basados en redes neuronales han mostrado recientemente su potencialidad en espectaculares casos de éxito: las victorias a humanos en el juego del *GO*, sistemas que superan el test de Turing, o el desarrollo de arte psicodélico, son algunos de los ejemplos más actuales [**?**]. Sin embargo, los recursos computacionales requeridos, así como el consumo de potencia y tamaño del hardware necesario para este tipo de sistemas han impedido hasta ahora su incorporación en dispositivos comerciales.

Por otro lado, el desarrollo microelectrónico de sistemas de procesamiento de las últimas décadas ha supuesto su implantación ubicua, dando lugar incluso a una nueva disciplina como son los sistemas ciber-físicos. A esta proliferación no ha sido ajeno el desarrollo de módulos de procesamiento basados en redes neuronales, con diseños para su incorporación específica a dispositivos portátiles con alimentación por baterías, como los smartphones, o con recursos energéticos limitados, como los vehículos auto-dirigidos. En general estos módulos neuronales de propósito específico, pensados para procesamiento y reconocimiento de imágenes, muestran unos niveles de consumo reducidos, del orden de cientos de mili-watts.

La potencialidad de las redes neuronales artificiales, capaces de establecer relaciones funcionales entre conjuntos de patrones de estímulos-respuestas sin un conocimiento previo matemático del problema a resolver, así como su capacidad de modificar dicha relación funcional de manera dinámica ante cambios en los patrones, hacen de estas técnicas de procesado una herramienta de gran flexibilidad. Para su inclusión en módulos de bajo consumo, como los nodos en una red inalámbrica de sensores, donde la vida operativa debe contarse en meses, es preciso reducir aún más el consumo y el tamaño de los dispositivos, procurando mantener su adaptabilidad. En este sentido, existen dos aproximaciones para su implementación: mediante procesadores digitales o con el diseño de sistemas de procesamiento analógicos o mixtos [**?**]. La elección de uno u otro método vendrá condicionada por diversos parámetros, entre los que se incluyen las restricciones de consumo y área, así como la resolución requerida en la respuesta del sistema.

En general, si las restricciones vienen impuestas por el consumo y área, las implementaciones analógicas integradas de los procesadores neuronales ofrecen claras ventajas. Si además es preciso disponer de la flexibilidad que supone la reprogramación de los parámetros de la red neuronal, una implementación mixta que incluya procesadores analógicos con almacenamiento digital puede ser la solución óptima. Un ejemplo de este tipo de aplicaciones puede ser el diseño de módulos de acondicionado y calibrado integrados para *smart sensors* [**?**], donde un sistema neuronal analógico permitiría optimizar la respuesta del sensor antes de su digitalización, compensando derivas y linealizándola, con la capacidad de adecuar su respuesta ante cambios en el comportamiento del propio sensor asociados al envejecimiento.

Algunas características de las redes neuronales como el aprendizaje adaptativo, la auto-organización y la tolerancia a fallos [**?**], las hacen sumamente atractivas para la resolución de problemas relacionados con la predicción de eventos,

clasificación y ajuste de datos, entre otros. Además, las redes implementadas en hardware presentan robustez a variaciones de proceso debido al esquema de retroalimentación inherente proporcionado por el mecanismo de aprendizaje de retropropagación de errores, teniendo en cuenta las no-idealidades de los circuitos, como sucede con el enfoque de aprendizaje *chip-on-the-loop* [**?**], el cual lleva a cabo el entrenamiento de la red fuera del chip pero teniendo en cuenta toda la circuitería utilizada en la implementación de los bloques característicos de la neurona.

El objetivo de este trabajo es generar modelos de alto nivel de los principales bloques que conforman una neurona diseñada de forma analógica, para su posterior inserción y simulación en distintas estructuras de red. El artículo está dividido de la siguiente forma: en la Sección 2 se presenta la estructura básica de una neurona y los principales bloques que la conforman; además se describe su implementación electrónica y se presentan algunos resultados obtenidos mediante simulación. En la Sección 3 se describe el modelo matemático de alto nivel generado para cada uno de los bloques y se hace una comparación de éste con la respuesta eléctrica. En la Sección 4 se presenta una implementación sencilla de ANN para verificar los modelos y, finalmente, en la Sección 5 se presentan las conclusiones del trabajo.

2. Implementación analógica de la neurona

Como es sabido, las neuronas son las unidades básicas de procesamiento de este tipo de sistemas, ya que se encargan de llevar a cabo todo el procesamiento de las señales. Es por esto que se debe tener especial cuidado en su diseño y modelado, sobre todo si se trata de una implementación en hardware.

En la implementación de la neurona es posible distinguir dos bloques significativos: un bloque de multiplicación encargado de multiplicar la señales de entrada o bien las señales de las capas intermedias, por un conjunto de pesos o coeficientes de ajuste, que pueden seer representados de manera analógica o digital; y un bloque que genera la función de activación para implementar la

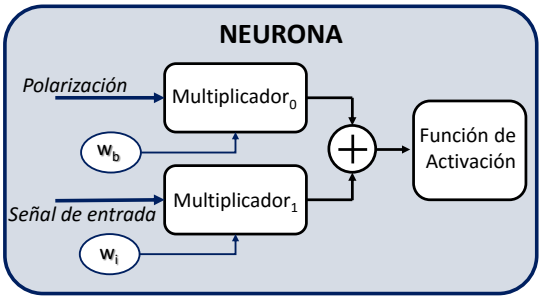


Fig. 1. Diagrama a bloques de la unidad básica de procesamiento

operación no-lineal. En la Fig. 1 está representado el esquema de esta unidad básica de procesamiento.

Es conveniente mencionar que para la implementación analógica de este tipo de sistemas, la tecnología CMOS es la opción más viable debido a que es una tecnología madura en cuanto a implementaciones VLSI, además de que ofrece la posibilidad de co-integración del sensor con la electrónica asociada a éste si fuera requerido. En el trabajo que se está desarrollando se utiliza una tecnología CMOS estándar de 0.18μm con voltaje de alimentación $V_{DD} = 1{,}8V$.

2.1. Multiplicador

Como ya se ha comentado, el multiplicador se encarga de ponderar una señal a partir de un coeficiente o peso de activación w. Éste sólo tomará valores comprendidos entre [-1,1], de tal forma que si $w = 0$ se entiende que la señal se inhibe y es cero, si $w = 1$, toda la señal se transmite, y si $w = -1$, la señal se invierte o cambia de signo.

El bloque implementado es un amplificador de ganancia programable (PGA) que actúa como multiplicador analógico-digital. Todo el procesamiento de la señal se hace en el dominio analógico, pero mediante un control digital se establece el factor de ponderación w. Este control permite definir de forma sencilla el valor del peso mediante una palabra digital.

La implementación del PGA se muestra en la Fig. 2. El circuito es un amplificador de voltaje que utiliza resistencias y transistores MOS para establecer la ganancia. Los dos transistores trabajan como como un divisor de corriente MOS (MCD) [**?**], en donde las corrientes I_1 e I_2 se relacionan de la siguiente forma:

$$I_1 = K_1[f_1(\psi_{sL_1}) - f_1(\psi_{s0_1})] = -V_{in}/R_1, \tag{1}$$

$$I_2 = K_2[f_2(\psi_{sL_2}) - f_2(\psi_{s0_2})] = -V_o/R_2. \tag{2}$$

Los potenciales de superficie de drenaje ψ_{sL_1} y ψ_{sL_2} son los mismos por conexión, mientras que los potenciales de superficie de fuente ψ_{s0_1} y ψ_{s0_2} son iguales por las condiciones de operación, ya que ambas fuentes están conectadas a la terminal de entrada de un amplificador, y los voltajes de compuerta y de

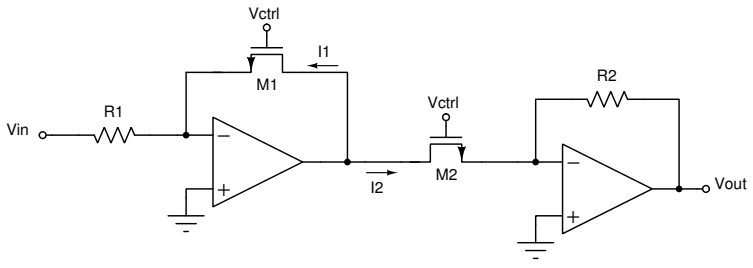


Fig. 2. PGA basado en divisores de corriente MOS

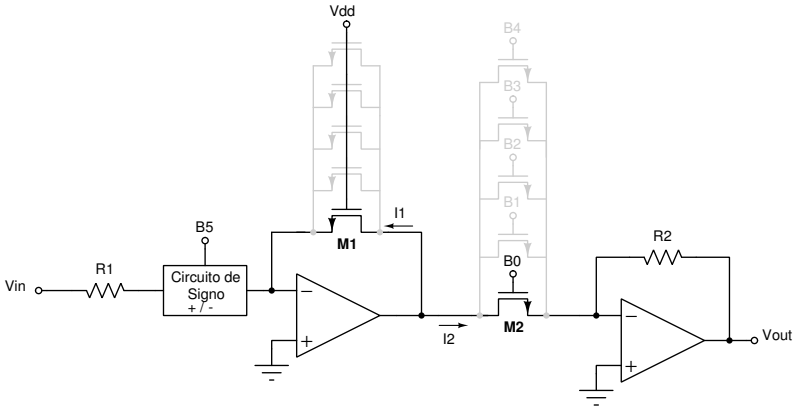


Fig. 3. Implementación eléctrica del bloque de multiplicación

cuerpo son iguales para ambos transistores. Por lo tanto, si M_1 y M_2 tienen un buen *matching*, la función de transferencia del amplificador va a estar dada por:

$$\frac{V_o}{V_{in}} = \frac{R_2}{R_1}\frac{K_2}{K_1} = \frac{R_2}{R_1}\frac{(W/L)_2}{(W/L)_1}. \tag{3}$$

A partir de esta expresión, el factor de proporcionalidad entre ambos transistores puede ser utilizado para definir la ganancia del amplificador y, por lo tanto, el factor de ponderación requerido. La manera más sencilla de hacerlo es utilizando un conjunto de transistores en paralelo como se aprecia en la Fig. 3, de tal forma que el transistor MOS equivalente generado tendrá una anchura igual a xW, donde x es el número de transistores en paralelo.

El esquema completo del multiplicador se muestra en la Fig. 3. La ponderación de la señal se controla mediante una palabra digital de 5 bits ($B_4...B_0$), haciendo posible tener 32 posibles valores entre 0 y 1 con un delta de variación entre cada bit igual a:

$$\Delta = \frac{1}{2^n - 1} = 32{,}258 \times 10^{-3}, \tag{4}$$

donde n es el número de bits. Ademas se ha añadido un bloque de signo, cuyo objetivo es cambiar el sentido de la señal que entra al PGA, de forma que sea posible tener también pesos negativos. Este circuito está controlado por el bit de signo B_5. Debido a que la máxima ganancia es 1, los transistores que conforman M_1 siempre están encendidos y operando en triodo, mientras que aquellos que conforman M_2 estarán controlados por la palabra digital. Así, si se desea la máxima ganancia $G = 1$, todos los bits de esta palabra digital deben ser V_{DD}.

Circuito de Signo. En la Fig. 4 se presenta el circuito de signo implementado. Este corresponde a un espejo de corriente clase AB [**?**]. En condiciones estáticas,

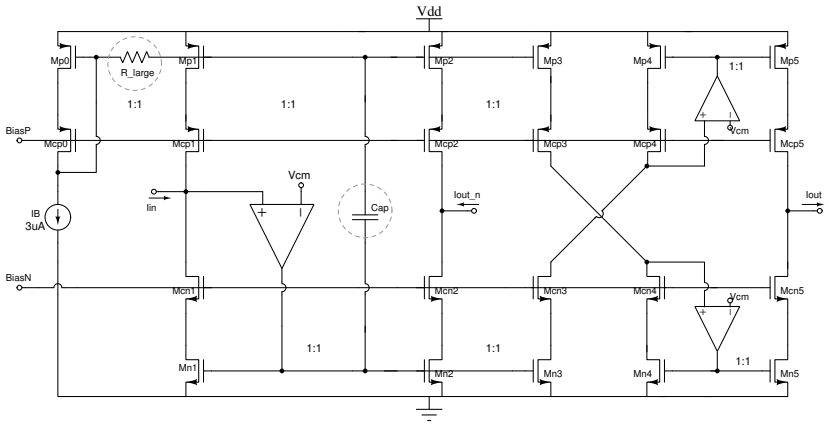


Fig. 4. Implementación eléctrica del bloque de signo

el capacitor se comporta como un circuito abierto y el esquema es equivalente a un espejo de corriente simple. Bajo condiciones dinámicas, la configuración opera en clase AB transformando los transistores M_{p1} y M_{p2} en fuentes de corriente dinámicas. El circuito cuenta con dos ramas de salida, una que pasa la corriente en la misma dirección en la que entra, y la otra que invierte su sentido. Mediante el bit B_5 es posible seleccionar qué rama se encuentra activa. El error relativo en la transferencia de corriente en ambas salidas es menor al 1 % considerando una corriente mínima de entrada $I_{in} = 10nA$; y la distorsión armónica total (THD) de la señal es menor a $-80dB$ teniendo en cuenta una señal máxima de entrada $I_{in} = \pm 15\mu A$ a una frecuencia de $1kHz$.

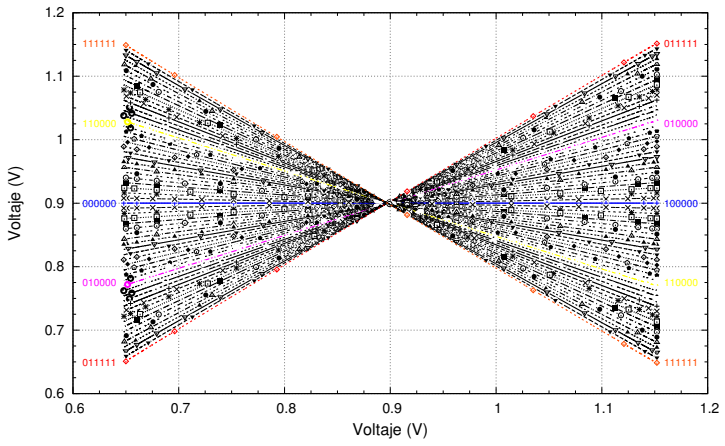
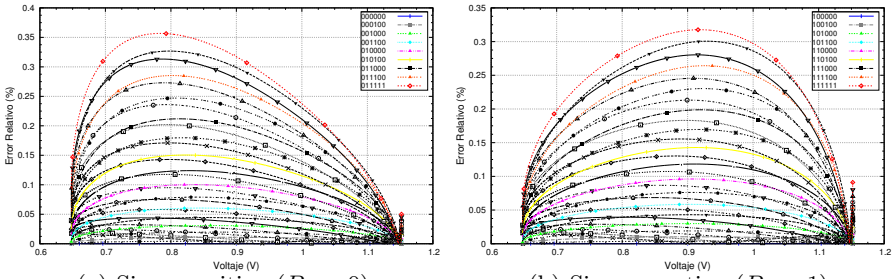


Fig. 5. Respuesta eléctrica del multiplicador en modo voltaje de 5 bits

Simulación del multiplicador. El voltaje de entrada considerado tiene un nivel de modo común $V_{cm} = 0{,}9V$. Señales de voltaje menores a este valor son consideradas negativas, mientras que por encima de $0{,}9V$ son positivas. Se simuló el multiplicador de la Fig. 3 considerando 5 bits más el bit de signo. El transistor correspondiente a cada bit está formado a su vez por un conjunto de transistores con una única dimensión. El amplificador utilizado en el esquema completo es un amplificador operacional de transconductancia (OTA) con etapa de salida clase AB con ganancia $G_{opam} = 76dB$ y resistencia de salida $R_{out} = 160\Omega$. Los valores de las resistencias son $R_1 = R_2 = 18k\Omega$.



(a) Signo positivo ($B_5 = 0$). (b) Signo negativo ($B_5 = 1$).

Fig. 6. Error relativo e_r con respecto a la respuesta calculada idealmente

La simulación se llevó a cabo utilizando el software de diseño electrónico CADENCE . En la Fig. 5 se representa el voltaje de salida con respecto al voltaje de entrada para las combinaciones posibles de bits. Se calculó el error relativo e_r que existe entre cada una de estas respuestas con respecto a la respuesta esperada o salida ideal. Para esto, se evaluó la expresión mostrada a continuación (Eq. 5) para calcular y graficar cada una de las respuestas ideales:

$$V_{out_{ideal}} = 0{,}9 - 0{,}9 \cdot w + V_{in} \cdot w, \tag{5}$$

donde V_{in} es el voltaje de entrada con un nivel de DC igual a $0{,}9V$, y w el peso.

En la Fig. 6 se presenta el error relativo calculado para cada palabra digital. El inciso (a) muestra el error relativo considerando sólo pesos positivos, es decir, sin inversión de la señal ($B_5 = 0$); mientras que en el inciso (b) se presenta el error relativo considerando inversión en la señal, y por lo tanto, pesos negativos ($B_5 = 1$). El máximo error relativo para el primer caso es $e_r = 0{,}3567\,\%$, y corresponde a la máxima ponderación de la señal ($w = 1$). Igualmente, tomando en cuenta sólo pesos negativos, el máximo error relativo calculado se tiene para la máxima ganancia y es $e_r = 0{,}3176\,\%$. En condiciones estáticas, el multiplicador presenta un consumo máximo de potencia $P_{max} = 42{,}8\mu W$, que se tiene cuando el bit de signo es '1'.

De acuerdo al diagrama de bloques de la neurona (Fig. 1), es necesario hacer una sumatoria de las señales ponderadas. Utilizando el mismo esquema

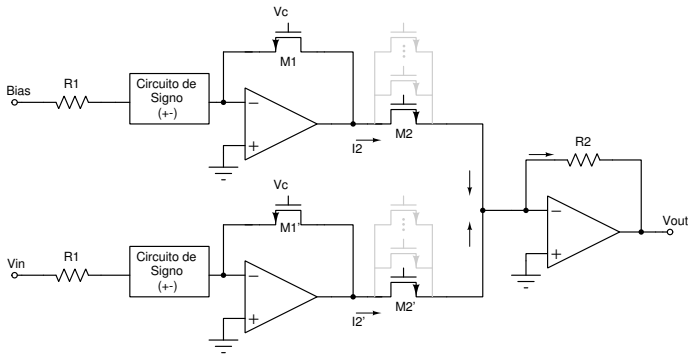


Fig. 7. Esquema para realizar la suma de las señales ponderadas

de multiplicación que se ha descrito, es posible realizar la suma de las señales en un nodo común, tal como se muestra en la Fig. 7. Las señales en corriente son sumadas en el nodo negativo del segundo amplificador, donde se tiene una tierra virtual debido a la configuración del circuito.

2.2. Función de activación

La función de activación calcula el estado de actividad de una neurona. Transforma la entrada global en un valor o estado de activación, cuyo rango normalmente es de [0,1] o [–1,1], ya que una neurona puede estar totalmente inactiva o activa. Las funciones de activación más utilizadas son: la función lineal, función escalón y función sigmoidal.

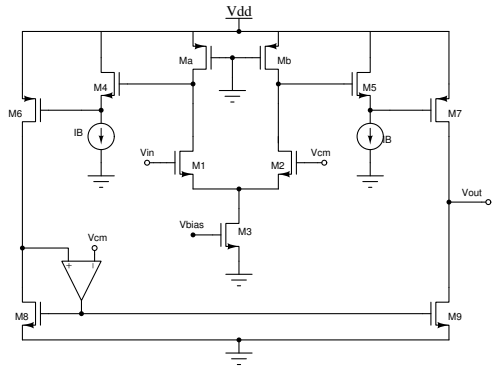
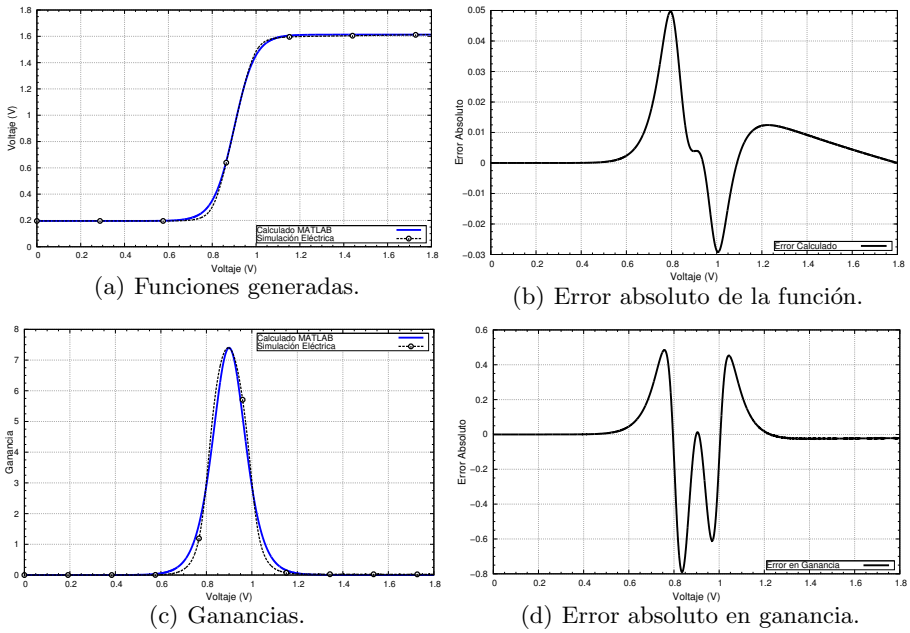


Fig. 8. Implementación eléctrica del bloque que genera la función de activación

En el trabajo se ha implementado una función sigmoide debido a que es diferenciable y presenta una característica no-lineal, propiedades importantes



(a) Funciones generadas.

(b) Error absoluto de la función.

(c) Ganancias.

(d) Error absoluto en ganancia.

Fig. 9. Comparación de la función de activación eléctrica con la función calculada

para el entrenamiento de redes multi-capa con retropropagación. La función implementada presenta una característica similar a una tangente hiperbólica, la cual tiene forma de 'S' con umbrales de saturación en -1 y $+1$, y pendiente no-lineal en la parte central. Sin embargo, en el circuito implementado estos niveles de saturación están definidos entre $0{,}2V$ y $1{,}6V$ con el mismo nivel de modo común que el multiplicador.

En la Fig. 8 se muestra el circuito eléctrico implementado. Está formado por un par diferencial con transistores NMOS como entradas, y cargas resistivas implementadas con los transistores M_a y M_b operando en la región lineal. El amplificador utilizado en el esquema fija el modo común $V_{cm} = 0{,}9V$ a la salida del circuito, y está implementado mediante un par diferencial sencillo. La resistencia equivalente de los transistores M_a y M_b es $R_{on} = 4{,}6k\Omega$, y la ganancia total del circuito es $G = 7{,}4$.

Simulación del Circuito. El circuito se simuló variando el voltaje de entrada entre 0 y $1{,}8V$. En la Fig. 9(a), con línea negra punteada, se muestra la respuesta obtenida, acompañada también, en continua azul, por una función tangente hiperbólica generada matemáticamente en MATLAB para que tuviera los mismos valores de ganancia ($G = 7{,}4$) y niveles de saturación ($V_{max} = 1{,}6$ y $V_{min} = 0{,}2$) que la implementación eléctrica.

En la misma figura, pero inciso (b), está representado el error absoluto obtenido al comparar ambas funciones. Se aprecia un máximo error absoluto $e_{abs(f)} = 0{,}04V$ que corresponde a la secciones con más no-linealidad de la respuesta (codos de la función). Por otro lado, en en inciso (c) se presenta el comportamiento de la ganancia (derivada de la función) considerando el rango completo del voltaje de entrada, mientras que en el inciso (d) está representado el error absoluto en ganancia calculado. Se aprecia que el máximo error absoluto en ganancia es de $e_{abs(g)} = 0{,}79$. Finalmente, el consumo total de potencia estática es de $P_{max} = 27{,}88\mu W$.

3. Modelo de alto nivel de los circuitos eléctricos

3.1. Multiplicador

Con los datos obtenidos de las simulaciones eléctricas se generó un vector Y con todos los valores de salida posibles y una matriz X que incluía en una columna los valores de entrada y en otra, los pesos de activación. A partir del arreglo de datos generado, y mediante MATLAB se obtuvo un modelo matemático que fuera válido para todos los casos. Esto se llevó a cabo con ayuda de la función *regstats*, la cual realiza una regresión multilineal de las respuestas en Y.

El modelo matemático que describe la respuesta del multiplicador de 6 bits (5 bits y el signo) es el siguiente:

$$V_{out} = 0{,}90001 - (1{,}0298 \times 10^{-5}) \cdot V_{in} - (0{,}8969) \cdot w + (0{,}9983) \cdot V_{in} \cdot w. \quad (6)$$

Como se aprecia en esta ecuación, el modelo es muy parecido al modelo ideal establecido en la ecuación (5). Se evaluó el modelo obtenido y se comparó con los resultados eléctricos. En la Fig. 10 se presenta la gráfica obtenida para cada palabra digital considerando el mismo rango de voltaje de entrada que en la simulación eléctrica. Los resultados muestran que se tiene un error cuadrático medio $mse = 2{,}0641 \times 10^{-7}$ entre los valores esperados (obtenidos por simulación) y los calculados a partir del modelo matemático.

3.2. Función de activación

Al igual que el multiplicador, se requiere un modelo matemático que represente el comportamiento eléctrico del circuito. Aunque hay algunas aproximaciones basadas en ecuaciones racionales de polinomios de orden mayor que se pueden obtener en MATLAB, se decidió utilizar una tabla de valores, ya que la primera opción no era práctica, pues los umbrales no saturaban y el tiempo de procesamiento era elevado. Se verificó que el modelo descrito con la tabla de valores es 100 veces más rápido que el implementado con la expresión polinomial.

Cabe mencionar que a partir del conjunto de valores, se obtuvo la derivada de la función y, de igual forma, se generó una tabla con estos valores.

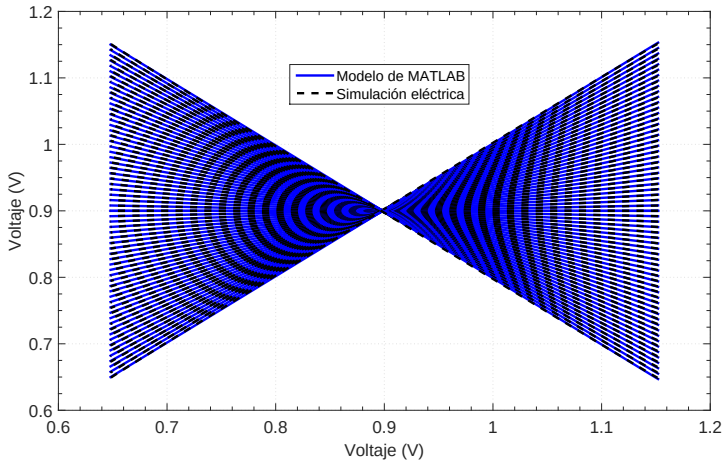


Fig. 10. Comparación de la respuesta obtenida con el modelo y la respuesta eléctrica

4. Resultados preliminares

El objetivo del trabajo, además de modelar el comportamiento eléctrico de los bloques característicos de la neurona, es comprobar que dichos modelos son funcionales al introducirlos en una red neuronal. En esta sección se presenta un ejemplo sencillo y práctico para verificar esto.

4.1. Solución del operador XOR

Una tarea sencilla, pero que requiere el uso de redes neuronales multicapa es la solución del operador OR exclusivo. Como se sabe, esta compuerta funciona de acuerdo a la lógica mostrada en la Tabla 1, donde la salida es igual a '1' cuando uno de los operandos es '1', pero no ambos.

Tabla 1. Tabla de verdad de la compuerta XOR

B	A	XOR
0	0	0
0	1	1
1	0	1
1	1	0

Con la herramienta *nntool* de MATLAB es posible crear, entrenar y simular redes neuronales de manera sencilla, y por esta razón es el software que se utiliza en el trabajo. Se creó una red neuronal multicapa de 2 entradas, 1 capa oculta con 4 neuronas y la capa de salida, tal como se muestra en la Fig. 11. En

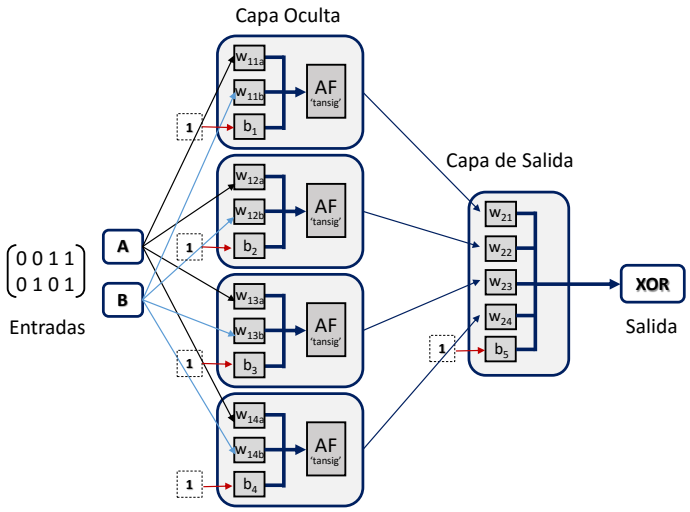


Fig. 11. Red neuronal creada en MATLAB para resolver el operador lógico XOR

esta se observa que hay 17 señales que necesitan ser ponderadas: en la capa oculta hay conexión de 8 señales que provienen de las entradas 'A' y 'B' y 4 señales de polarización o *bias*. Luego, en la última capa están conectadas las señales de salida de las 4 neuronas de la capa anterior y su respectiva señal de polarización. La función de activación en cada neurona de la capa oculta es la tangente hiperbólica.

El algoritmo de entrenamiento de la red está definido en la función 'TRAINLM'. Ésta corresponde al algoritmo *Levenberg-Marquardt* de retropropagación, el cual realiza las iteraciones y actualiza los pesos empleando el método de optimización por descenso de gradiente [**?**]. Aunque es posible utilizar otros algoritmos, éste es el más rápido y eficiente para una red de este tamaño.

Entrenamiento de la red con los modelos obtenidos. Es conveniente mencionar, que para el entrenamiento de la red, se estableció que el '1' lógico corresponde a un nivel de voltaje de $1{,}6V$, mientras que el '0' está definido como $0{,}2V$, ya que son los voltajes máximo y mínimo que maneja el bloque de la función de activación.

Entrenando la red con las funciones predefinidas de la herramienta, el sistema resuelve el problema en 5 iteraciones, y se observa que los valores de los pesos de activación resultantes toman cualquier valor positivo o negativo.

Sin embargo, como se ha establecido en el trabajo, el rango de variación de los coeficientes de activación debe estar limitado entre -1 y 1, ya que así se ha establecido en el multiplicador diseñado. Además, debido a que la ganancia del PGA se controla con 5 bits, y recordando la ecuación (4), la resolución que puede manejar también está limitada a $32{,}258 \times 10^{-3}$. Para llevar a cabo el

entrenamiento tomando en cuenta los bloques diseñados y sus condiciones de operación, es necesario introducir sus respectivos modelos en el esquema de red generado y modificar el algoritmo de entrenamiento para que considere estas condiciones.

En el caso del multiplicador, normalmente al definir la red la estructura emplea la función *producto*, que está definida como `z=w*p`, donde w hace referencia a la matriz de pesos y p a las señales. Para utilizar el bloque diseñado, se requiere cambiar esta función por el modelo matemático generado. También es necesario obtener la derivada de la expresión con respecto a w y la derivada con respecto a p (señal o entrada), pues son necesarias en el algoritmo de entrenamiento. De esta forma, dentro del toolkit *nnet*, se generó una nueva función definiendo la multiplicación y las derivadas de la siguiente forma:

```
z     = 0.900013675640531-(1.02981131896217e-05)*p
        -(0.896976394158687)*w + (0.998246762810866)*w*p;
dz_dw = - 0.896976394158687 + 0.998246762810866*p;
dz_dp = - 1.02981131896217e-05 + 0.998246762810866*w.
```

La función de activación predefinida en la estructura de red generada es la tangente hiperbólica o *tansig*. De igual forma que el multiplicador, se generó una nueva función en donde se definió el modelo matemático del circuito eléctrico. Al tratarse de una tabla, simplemente se programó para que fuera capaz de direccionar a la posición adecuada. Para el caso de la derivada, el procedimiento es similar.

Con los nuevos modelos del multiplicador y de la función de activación, dentro de la estructura de red se cambiaron las funciones predefinidas por los nuevos modelos, de forma que en todos los puntos donde se utiliza el operador producto, ahora se va a utilizar la nueva función (igual para la función no-lineal). Además, se incluyeron algunas líneas de código en la función que realiza el entrenamiento (*TRAINLM*) para delimitar el rango de variación de los pesos y también para discretizarlos, de manera que sólo tome valores que puedan ser generados con los 5 bits considerados.

Con las modificaciones mencionadas se entrenó la red neuronal. Después de 8 iteraciones, la red llegó a la solución y estableció los valores de los 17 pesos. En el circuito eléctrico, éstos se interpretan como las ganancias que debe tener cada multiplicador (PGA). En la Tabla **??** se presenta el valor de cada uno de los coeficientes junto con su identificador (definido en la Fig. 11), así como su correspondiente palabra digital.

Finalmente, al simular la red con estos coeficientes, la salida corresponde con los valores esperados:

```
out = [0.26682 1.5262 1.5352 0.25703].
```

Tabla 2. Pesos de activación obtenidos al finalizar el entrenamiento de la red

Peso (w)	Valor	Palabra Digital	Peso (w)	Valor	Palabra Digital	Peso (w)	Valor	Palabra Digital
w_{11a}	-0.29032	101001	w_{11b}	0.41935	001101	b_1	-0.48387	101111
w_{12a}	-0.77419	111000	w_{12b}	-0.45161	101110	b_2	-0.29032	101001
w_{13a}	-0.32258	101010	w_{13b}	0.80645	011001	b_3	0.3871	001100
w_{14a}	-0.48387	101111	w_{14b}	0.48387	001111	b_4	0.67742	010101
w_{21}	0.74194	010111	w_{22}	1	011111	b_5	-0.3871	101100
w_{23}	1	011111	w_{24}	-1	111111	–	–	–

5. Conclusiones

En este trabajo se ha presentado la implementación analógica de los bloques característicos que conforman la unidad de procesamiento básica de una red neuronal, así como su modelado matemático en MATLAB.

El diseño electrónico del multiplicador y del bloque que genera la función no-lineal se realizó en tecnología CMOS de 0.18μm con alimentación de 1.8V. Las simulaciones eléctricas de ambos bloques muestran buenos resultados al compararlas con funciones ideales implementadas numéricamente, ya que, para el caso del multiplicador de 6 bits (5 para definir el peso y 1 para controlar el signo) implementado, el error relativo e_r se mantiene por debajo del 1 %, mientras que para el caso del bloque no-lineal con característica de tangente hiperbólica, el máximo error absoluto es $e_{abs} = 0{,}04V$.

Además, con los datos de simulación, se modeló en alto nivel el comportamiento eléctrico de ambos bloques y se verificó, resolviendo el operador lógico XOR, que es posible entrenar una red neuronal introduciendo dichos modelos en la estructura de ésta. Así mismo, modificando el algoritmo de entrenamiento es posible discretizar y delimitar los valores posibles de los coeficientes de activación.

Agradecimientos. Este trabajo ha sido financiado por CONACYT con el número de beca de doctorado 362674, así como por el proyecto de investigación MINECO-FEDER con número TEC2015-65750-R.

Referencias

1. Bourzac, Katherine: Neural Networks on the Go. IEEE Spectrum (2016)
2. Draghici, S.: Neural Networks in Analog Hardware - Design and Implementation Issues. International Journal of Neural Systems, vol. 10, no. 1, 19–42 (2000)
3. Horn, G.van der J., Huijsing, L.: Integrated Smart Sensors: Design and Calibration. Kluwer Academic Publishers (1998)
4. Graude, D.: Principles of Artificial Neural Networks. Advanced Series on Circuits and Systems, vol. 6., 2nd. Edition. World Scientific Publishing Company (2007)

5. Sanz, M. T., Celma, S., Calvo, B.: Using MOS Current Dividers for Linearization of Programmable Gain Amplifiers. International Journal of Circuit Theory and Applications, vol. 36, no. 4, 397–408 (2008)
6. Lopez-Martin, A.J., Ramirez-Angulo, J., Carvajal, R.G., Algueta, J.M.: Compact Class AB CMOS Current Mirror. Electronics Letters, vol. 44, no. 23, 1335–1336 (2008)
7. Beale, M. H., Hagan, M. T., Demuth, H. B.: MATLAB Neural Network Toolbox: User's Guide. The MathWorks, Inc. (2015)

Clasificación de patrones mediante el uso de una red neuronal pulsante

Christian Hernández-Becerra, Manuel Mejía-Lavalle

Centro Nacional de Investigación y desarrollo Tecnológico,
Departamento de Ciencias Computacionales, Cuernavaca, Morelos,
México
{chrishb, mlavalle}@cenidet.edu.mx

Resumen. Se muestra cómo, mediante el uso de una sola capa de neuronas pulsantes, más aún con una sola neurona, es posible hacer la clasificación de patrones, ya sea de una función binaria como la función XOR o bien de una base de datos con decenas de características. Se ocupa el modelo de *Izhikevich* para modelar el comportamiento de las neuronas pulsantes utilizadas. Principalmente se pretende explotar el uso de una sola neurona para lograr realizar clasificación, analizando el posible alcance. Los resultados obtenidos son alentadores.

Palabras clave: Redes neuronales pulsantes, neurona de *Izhikevich*, clasificación, función XOR, base de datos.

Patterns Classification Using Spiking Neural Networks

Abstract. It is shown how, through the use of a single layer of spiking neurons, even more so with a single neuron, it is possible to make the classification of patterns, either a binary function as the XOR function or a database with tens of characteristics. *Izhikevich* model is used for modeling the behavior of the used spiking neurons. Mainly intends to exploit the use of a single neuron to perform classification, analyzing the possible scope. Obtained results are encouraging.

Keywords: Spiking neural networks, *Izhikevich* neuron, classification, XOR function, database.

1. Introducción

El proceso de clasificación y reconocimiento son un par de las características del ser humano que evidentemente han jugado un papel importante en su evolución. Resulta de interés para la Inteligencia Artificial poder recrear estas importantes habilidades del ser humano de forma minuciosa.

Teniendo en cuenta que estos procesos se llevan a cabo en el cerebro, no resulta descabellado ocupar herramientas como las Redes Neuronales Artificiales (RNAs) como principal herramienta, dado que los procesos cognitivos del ser humano (así como el de los mamíferos en general) suceden en el cerebro mediante la comunicación de las neuronas.

Uno de los modelos que surgen de las RNAs son los conformados por neuronas denominadas de tercera generación [1] cuya principal característica es la similitud que tienen con las neuronas que conforman el cerebro desde la perspectiva biológica. De forma más específica, las neuronas artificiales de este tipo simulan un voltaje de membrana y, al igual que las neuronas biológicas, arrojan una cantidad de pulsos a lo largo del tiempo. Es mediante estos pulsos que las neuronas se comunican y realizan determinados procesos.

1.1. Redes neuronales

Se pueden distinguir fácilmente tres posibles generaciones las RNAs. La primera o llamada Perceptron, basada en el modelo de McCulloch-Pitts[2], como unidades computacionales (0 y 1). El rasgo característico es que solo pueden dar como salida un dígito. Sin embargo cada función booleana se puede calcular por algunas multicapas de Perceptron con una sola capa oculta. La segunda generación se basa en el mismo modelo pero aplican una "función de activación"de imagen continua a una combinación de las entradas, la más común es la función sigmoidea[3]. Finalmente las de tercera generación son las llamadas pulsantes, que mediante una descripción matemática modelan con más realismo las neuronas biológicas [1].

1.2. Modelo *Izhikevich*

Existen varias estructuras de ecuaciones matemáticas que pretenden modelar el comportamiento de las neuronas artificiales pulsantes en respuesta al voltaje que reciben: Integrate-and-fire, FitzHugh-Nagumo, HindMarsh- Rose, Hoodgkin-Huxley, *Izhikevich*, entre otros [4].

En la referencia [4] se exponen dichos modelos, así como una comparación entre ellos para aportar un argumento sobre cuál de los modelos conviene utilizarse en términos de sus características. La Figura 1 ofrece una forma de comparación sencilla entre los modelos abordados en [4].

De esta manera es que se decide ocupar el modelo de *Izhikevich*, el cual se conforma de las Ecuaciones 1 y 2, y la condición de disparo de la Ecuación 3:

$$C\frac{dv}{dt} = k(v - v_r)(v - v_t) - u + I, \tag{1}$$

$$\frac{du}{dt} = a(b(v - v_r) - u), \tag{2}$$

$$\text{Si } v \geq v_{peak} \;\Rightarrow\; v = c,\; u = u + d. \tag{3}$$

De dichas ecuaciones se tienen las variables de la Tabla 1.

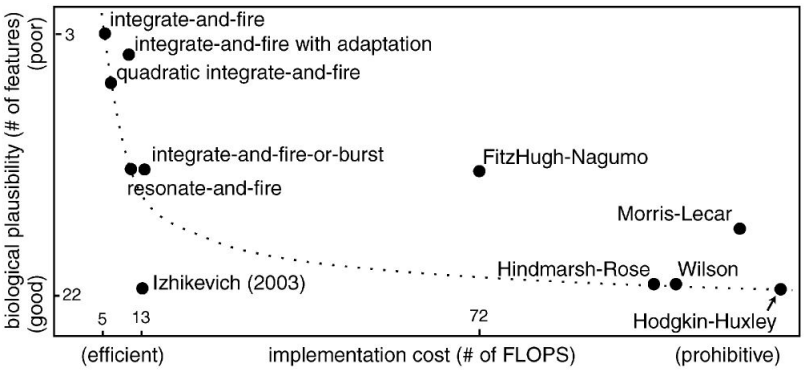


Fig. 1. Comparación entre la verosimilitud biológica y la eficiencia en términos de la fidelidad al comportamiento biológico y el costo de implementación [4]. Se observa que *Izhikevich* es la mejor opción

Como es conocido, una de las partes fundamentales de las RNAs es el entrenamiento de la red. Es importante mencionar que a diferencia de los modelos de primera y segunda generación, es ligeramente más difícil realizar el entrenamiento de la red por métodos tradicionales, lo que resulta es que en el modelo de neuronas de tercera generación, se ocupan otros métodos. Mas adelante se explica qué herramientas se usan para realizar el proceso de entrenamiento.

La estructura del resto del artículo es la siguiente: en la Sección 2 se muestran las herramientas que son ocupadas a lo largo de esta publicación. En la Sección 3 se muestra el desarrollo para realizar la clasificación de la función XOR con una sola neurona. En la Sección 4 se muestra el desarrollo para realizar la clasificación de base de datos *Ionosphere* con una sola capa de neurona. Finalmente en la Sección 5 se concluye y se discuten trabajos futuros.

2. Descripcción de las herramientas ocupadas

Parte de lo que se pretende con este trabajo es mostrar un poco del alcance que tienen las neuronas pulsantes dentro de la clasificación por sí solas, es decir, de cierta forma se pretende vislumbrar qué tanto se puede lograr con un mínimo de neuronas y tener una idea más clara del poder que tiene dicha herramienta.

2.1. Una capa

Aun cuando una RNA consta generalmente una capa de entrada, una o varias capas ocultas y una capa de salida [5], es posible lograr la clasificación con una sola capa cuando se trata de un problema de clasificación simple. Sin embargo cuando el problema aumenta de complejidad, una sola capa de neuronas de primera o segunda generación puede complicar la interpretación de la solución.

Tabla 1. Descripción de las Variables de las Ecuaciones 1, 2 y 3

Variable	Significado
I	Voltaje de entrada a la neurona
a	Constante del tiempo de recuperación
b	Constante de sensibilidad de la neurona
u	Variable de recuperación
C	Capacitancia
d	Variable de restablecimiento después del disparo
v_r	Valor de voltaje de la neurona en reposo
v_t	Voltaje del umbral instantáneo
c	Voltaje de reinicio
v	Potencial de la neurona
k	Parámetro para la forma del pico
v_{peak}	Valor del umbral de disparo

Aquí se ocupará una sola capa de neuronas pulsantes donde la entrada estará codificada por una función I definida en términos de las características de los patrones, y la salida se representa mediante la cantidad de disparos que suceden cuando el voltaje de la neurona supera el umbral v_{peak}, esta variable la llamaremos s.

Pese a que se podría escoger una cantidad arbitraria de neuronas para la capa, se realizan las pruebas con el mínimo posible, es decir, una neurona.

2.2. Modelo *Izhikevich*

Para el modelo de la neurona se ocupan los valores sugeridos en [7] y en [6] que se muestran en la Figura 2 para recrear disparos tipo *chattering*.

En la Tabla 2 se muestran los valores que se visualizan en la Figura 2 se ocupan para describir los picos mencionados.

Hay tres variables que aparecen en la Tabla 1 y que no aparecen en la Tabla 2, a saber: I, el voltaje de la neurona; v potencial de neurona y u, el voltaje de recuperación. Para poder ingresar información (voltaje) a la neurona, cada patrón debe ser convertido en un voltaje de entrada por medio de una función I que se describe en la siguiente subsección. El voltaje de recuperación es innecesario para los efectos del cómputo de los pulsos de respuesta y el voltaje v es precisamente donde se cuenta las veces que alcanza el valor v_{peak}.

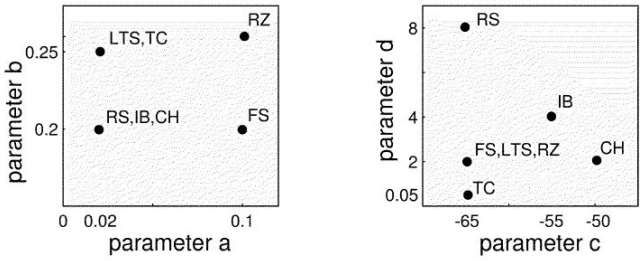


Fig. 2. Gráfica de los parámetros descritos en [7] y [6]. Los parámetros que interesan en este trabajo son lo que describen picos tipo *chattering*

Tabla 2. Valores de las variables del modelo de *Izhikevich*

Variable	Significado	Variable	Significado
a	= 0.03	v_t	= -40
b	= -2	c	= -50
C	= 100	k	= 0.7
d	= 100	v_{peak}	= 35
v_r	= -60		

2.3. Función de voltaje *I*

Algunas de las principales funciones que se ocupan para la codificación del voltaje se muestran en las Ecuaciones 4, 5 y 6. Sin embargo la que se utilizará en la experimentación es la función polinomial 4. En las tres ecuaciones, el valor de θ es el umbral mínimo que necesita la neurona para dar un disparo, pues no siempre es suficiente la codificación para que se logren dar pulsos [1].

Función polinomial. Es la función dada por la Ecuación:

$$I = (x \cdot w' + 1)^p + \theta. \tag{4}$$

Función productos. Es la función dada por la Ecuación:

$$I = (x \cdot w' \cdot \gamma) + \theta. \tag{5}$$

Función gaussiana. Es la función dada por la Ecuación:

$$I = e^{\frac{-||x-w'||^2}{2\sigma 2}} + \theta. \tag{6}$$

2.4. Algoritmo evolución diferencial

Este algoritmo cuya descripción detallada se encuentra en [9], es usado como una herramienta de aprendizaje no supervisado en [8]. De forma similar se ocupa aquí.

Como es conocido, para un algoritmo evolutivo se requiere una población, para este problema la población consiste de vectores conformados por pesos de la neurona. Es decir, cada elemento de la población del algoritmo evolutivo se interpreta como los pesos de red neuronal pulsante de una neurona. El algoritmo se describe en los párrafos siguientes.

El individuo i-ésimo de la generación G está denotado por $x_{i,G}$ y donde i es un valor entre 1 y NP, considerando que NP el numero de elementos en la población. En la primera generación los elementos de la población son creados de forma aleatoria (en nuestro caso siguiendo una distribución uniforme entre 0 y 1). Partiendo de la generación G, para la nueva generación se crean vectores de mutación $v_{i,G+1}$ se la siguiente manera: se eligen aleatoriamente 3 elementos de la población de la generación G denotados por $x_{r_1,G}, x_{r_2,G}$ y $x_{r_3,G}$ en donde queda claro que $r_1, r_2, r_3 \in \{1, ..., NP\}$. Se ocupa también una constante de *amplificación de diferencia de variación* $F \in [0, 2]$, finalmente se genera el vector de mutación con la ecuación 7:

$$v_{i,G+1} = x_{r_1,G} + F \cdot (x_{r_2,G} - x_{r_3,G}). \tag{7}$$

Note que para cada elemento de la población i se está generando un vector de mutación, eso es por que realiza un cruzamiento entre el el vector $x_{i,G}$ y el vector $v_{i,G+1}$ decidiendo con una probabilidad de $C_r \in (0, 1)$ si se hace o no el cambio sobre cada componente de los vectores. El vector resultante del cruzamiento se denota por $u_{i,G+1}$. Finalmente la selección de los elementos $x_{i,G+1}$ que conforman la siguiente generación está definida por la función de costo mínimo, es decir, si el vector $u_{i,G+1}$ produce el valor menor que el vector $x_{i,G+1}$ en la función de costo mínimo, entonces $x_{i,G+1} = u_{i,G+1}$ y su no, entonces $x_{i,G+1} = x_{i,G}$. Para nuestro problema, la función busca el máximo porcentaje de clasificación de los elementos representantes.

3. Ejemplo de clasificación, función XOR

Uno de los problemas más conocidos de clasificación es el de la función XOR (o también conocido como OR exclusivo) el cual no es posible clasificar mediante una función lineal, es decir, con el Perceptron. Aquí se ocupa una sola neurona con 2 parámetros de entrada para realizar la clasificación. Como es conocido, la función $XOR : \{0, 1\} \times \{0, 1\} \to \{0, 1\}$ se define como lo muestra la Tabla 3 , y su representación gráfica se puede apreciar en la Figura 3.

Una forma de resolver este problema de clasificación con neuronas de primera y segunda generación es utilizando 3 neuronas divididas en tres capas (dos en la capa oculta y una en la capa de salida). La red neuronal de una capa que se propone aquí para resolver este problema de clasificación consta de una

Tabla 3. Descripción de la función XOR

x_1	x_2	Clasificación
1	1	1
1	0	0
0	1	0
0	0	1

sola neurona, con 2 entradas. Se ocuparon como patrones de entrenamiento los elementos de la Tabla 3.

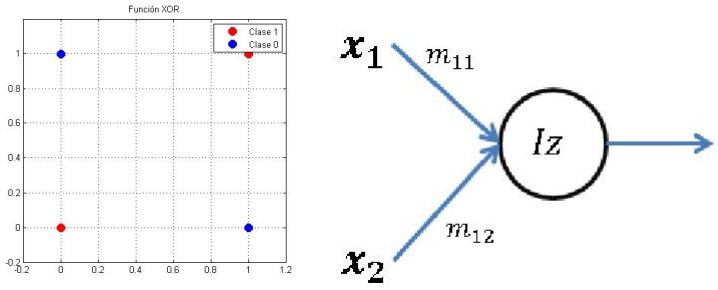


Fig. 3. Gráfica de la función XOR. Y la Neurona que se va a ocupar

Esta neurona se entrenó mediante el algoritmo de evolución diferencial. Dicho algoritmo, como ya se mencionó anteriormente, ocupa como función el porcentaje de elementos correctamente clasificados, es decir, que se realiza el entrenamiento buscando el 100 % de aciertos en la clasificación de lo patrones que se proporcionaron para entrenar la red (Los 4 elementos representados la Tabla 3).

La población para el algoritmo diferencial son vectores de dos dimensiones donde la componente i del vector corresponde al peso m_{1i}. Los valores m_{11} y m_{12} resultantes fueron en promedio (promedio de realizar 100 veces el mismo entrenamiento). En la Figura 4 se pueden ver los pulsos para los patrones del entrenamiento:

$$m_{11} = -0.9066 \quad \text{y} \quad m_{12} = -0.9089. \tag{8}$$

Finalmente con lo pesos arrojados por el algoritmo evolutivo mostrados en la ecuación 8 se tiene la red (conformada con una sola neurona) ya entrenada en donde la precisión de la clasificación es del 100 % de los elementos de entrenamiento.

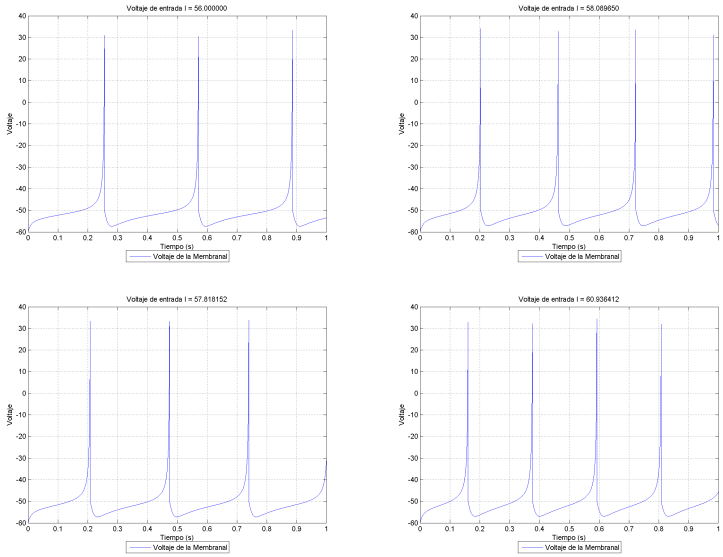


Fig. 4. Gráficas de pulsos de la neurona de *Izhikevich* para los elementos de entrenamiento

Para analizar la precisión de la clasificación, se puso a prueba la neurona con 400 patrones generados de forma aleatoria obtenidos aplicando una perturbación gaussiana sobre los patrones que se ocuparon para el entrenamiento, con una media de $\mu = 1$ ó 0 (si el valor es 1 ó 0 respectivamente), y varianza $\sigma = 0.1$. Se obtiene como resultado de clasificación un porcentaje del 97.5 % de patrones correctamente clasificados. La Figura 5 muestra la gráfica de la clasificación.

Con lo que se puede ver que la calidad de la clasificación es considerable dado que sólo se ocuparon 4 elementos para entrenar a la neurona. Evidentemente, si se proporcionan más elementos para el entrenamiento de la red neuronal, se logrará un porcentaje de clasificación más próximo al 100 %. En lo que sigue se realiza un análisis similar con una base de datos más compleja.

4. Datos de *Ionosphere*

4.1. Clasificación de *Ionosphere*

Se obtuvo la base de datos de *Ionosphere*.data que contiene información sobre electrones libres en la ionosfera. Los ecos de radar marcados como *buenos* son los que muestran evidencia de algún tipo de estructura en la ionosfera y los marcados como *malos* son aquellos que no. Estas señales tienen 17 pulsos y 2 atributos por cada impulso, por lo tanto cada instancia tiene 34 atributos. La base de datos contiene un número total de 351 instancias. En el atributo 35 se asigna la clasificación que puede ser *bueno* o *malo*. Con la descripción de la base

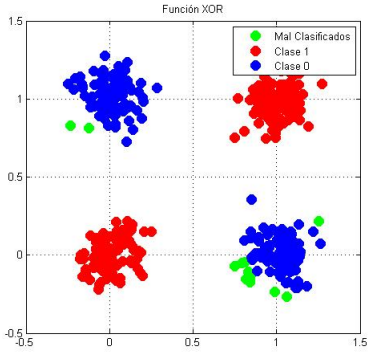


Fig. 5. Gráfica de factibilidad de clasificación de la neurona de *Izhikevich*

de datos se tiene que se ocupará una sola neurona con 34 parámetros de entrada para realizar la clasificación. Ver figura 6.

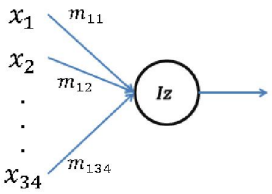


Fig. 6. Modelo gráfico de la neurona utilizada para la clasificación de los datos *Ionosphere*

Esta neurona se entrenó mediante el algoritmo de evolución diferencial de forma similar al problema de clasificación XOR, salvo por un paso intermedio. Como se puede observar, aquí no se cuenta con representantes, por lo que se seleccionó un porcentaje de la base de datos de forma aleatoria para realizar el entrenamiento. El porcentaje utilizado para el entrenamiento fue del 20 %. El algoritmo realizó el entrenamiento buscando el 100 % de aciertos de la población de muestra, es decir, de 70 elementos de la población total para el entrenamiento. Los parámetros del algoritmo evolutivo se describen en la Tabla 4.

Es importante notar que los parámetros del algoritmo diferencial se escogieron austeros para mostrar la simplicidad de condiciones necesarias. Los valores de los pesos que se encontraron con el algoritmo de entrenamiento permitieron una clasificación en promedio del 87.3 % en el conjunto de entrenamiento. En la Tabla 5 se detalla la cantidad promedio de disparos que se presentaron en las dos clases.

Tabla 4. Parámetros del algoritmo diferencial

Parámetro	Valor
Parámetro	Valor
Generaciones	$G = 100$
Total de la población	$NP = 40$
Tasa de diferencia de Variación	$F = 0.9$
Factor de recombinación	$C_r = 0.8$

Tabla 5. Resultados de la clasificación de *Ionosphere*

Clase	Medias de disparos (Representante de clasificación)
Malos	14.9682
Buenos	37.02222

Cuando se puso a prueba la red entrenada con los pesos encontrados mediante el algoritmo diferencial (que ya clasificaba correctamente el 87.3 % de los 70 elementos escogidos al azar para entrenar) se logró la clasificación correcta de 273 elementos de los 351, equivalente al 77.78 % de efectividad. Es decir que, en este caso, el 20 % de la población es suficiente para clasificar correctamente al 77.78 % de la población total.

5. Conclusiones y trabajo futuro

El desarrollo de una red multicapas es útil, pero en ocasiones hay problemas que pueden resolverse con menos "esfuerzo". Las neuronas pulsantes pueden ser ocupadas de forma muy simple en cuanto a (la estructura) obteniendo un resultado aceptable en la clasificación de patrones. Como se mostró en el presente trabajo, teniéndose resultados bastante alentadores. A lo largo del desarrollo de este trabajo se encontraron varias líneas que resultan interesantes y de las cuales se podrían obtener experimentos y observaciones con prometedoras conclusiones, por ejemplo en el problema de clasificación de la función XOR, resulta interesante indagar la forma de la separación del espacio de patrones cuando en el entrenamiento se presentan sólo los 4 representantes de la Tabla 3.

Es interesante analizar la clasificación lograda con la base de datos *Ionosphere*, y esto se podría hacer de forma similar en otras bases de datos. También

parece de interés averiguar qué tanto deben cambiarse los parámetros del algoritmo diferencial para lograr un mejor porcentaje de clasificación, o bien averiguar si hay algunos factores ajenos a la red neuronal que dificultan la clasificación (como diferentes subclases dentro de una misma clasificación).

Aparentemente, las neuronas pulsantes no ponen restricción a la clasificación binara (bueno o malo), es decir, que una línea de interés a futuro puede ser averiguar hasta dónde se puede clasificar con una sola neurona cuando se necesitan más de dos clases. Queda claro que el tema es muy basto y que aún cuando se descubren respuestas, siempre se encuentran nuevas preguntas.

Referencias

1. W. Maass: Networks of Spiking Neurons: The Third Generation of Neural Network Models. Institute for Theoretical Computer Science. Technische Universitit Graz, Neural Networks, Vol. 10, No. 9, pp. 1659–1671 (1997)
2. W. McCulloch, W. Pitts: A logical calculus of the ideas immanent in nervous activity. Bulletin of Mathematical Biophysics, 5:115–133 (1943)
3. D. E. Rumelhart, J. L. McClelland: Parallel distributed processing. Vol. 1, IEEE (1988)
4. E.M. Izhikevich: Which Model to Use for Cortical Spiking Neurons? IEEE Transactions On Neural Networks, Vol. 15, pp. 1063–1069 (2004)
5. W. Gerstner, W. Kistler: Spiking Neuron Models: Single Neurons, Populations, Plasticity. Cambridge University Press (2002)
6. E.M. Izhikevich: Simple Model of Spiking Neurons. IEEE Transactions on Neural Networks, Vol. 14, pp. 1569–1572 (2003)
7. I. C. Matadamas: Aplicación de las redes neuronales pulsantes en el reconocimiento de patrones y análisis de imágenes. Tesis de maestría. Centro de Investigación en Computación, Instituto Politécnico Nacional, México D.F. (2014)
8. R. Storn, K. Price: Differential Evolution-A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. Journal of Global Optimization, Vol. 11, pp. 341–359 (1997)
9. J. I. Espinosa-Ramos, N. Cruz-Cortes, R. A. Vázquez: Creation of Spiking Neuron Models Applied in Pattern Recognition Problems. In: Proceedings of International Joint Conference on Neural Networks, Dallas, Texas, USA (2013)
10. A. L. Hodgkin, A. F. Huxley: A quantitative description of membrane current and application to conduction and excitation in nerve. J. Physiol., vol. 117, pp. 500—544 (1954)
11. E. Izhikevich, G. M. Edelman: Large-Scale Model of Mammalian Thalamocortical Systems. PNAS, 105(9), pp. 3596–3598 (2008)

Mejora del desempeño de SVM usando un GA y la creación de puntos artificiales para conjuntos de datos no balanceados

José Hernández Santiago[1,2], Jair Cervantes Canales[2], Carlos Hiram Moreno Montiel[1], Beatriz Hernández Santiago[1]

[1] Tecnológico de Estudios Superiores de Chimalhuacán, Edo. de México, México

[2] Posgrado e Investigación Universidad Autónoma del Estado de México, Edo. de México, México

{jhernandezs, jcervantesc}@uaemex.mx, {josehernandez, carlosmoreno}@teschi.edu.mx

Resumen. En el mundo real los conjuntos de datos son regularmente no balanceados representando un problema crucial en Aprendizaje de Máquinas, ya que provoca una baja precisión en la mayoría de las técnicas de clasificación. Las SVM han reportado una excelente capacidad de generalización en los últimos años; sin embargo, al trabajar con conjuntos de datos no balanceados presentan un desempeño pobre debido a que el hiperplano obtenido queda sesgado hacia la clase mayoritaria. En este artículo, se presenta un nuevo algoritmo capaz de generar puntos artificiales dentro de la clase minoritaria y en la frontera entre clases a partir de los Vectores Soporte positivos, estos nuevos puntos son agregados al conjunto de entrenamiento a fin de disminuir el desbalance entre clases, mejorando el desempeño de las SVM en la mayoría de las pruebas.

Palabras clave: Máquinas de vectores soporte, conjuntos no balanceados, puntos artificiales, vectores soporte, sensitividad, especificidad, algoritmo genético.

Improved Performance of SVM Using a GA and the Creation of Artificial Points for Unbalanced Data Sets

Abstract. Real world data sets are regularly unbalanced, which is a crucial problem in Machine Learning, it causes a low accuracy in most classification techniques. SVM have reported excellent generalization capability in recent years; however, working with unbalanced data sets have poor performance due to the retrieved hyperplane is biased towards the majority class. This article presents a new algorithm capable of generating artificial points within the minority class and on the border between classes from the positive Support Vectors, these new points are added to the training data set in order to reduce the imbalance between classes, improving the performance of SVM in the majority of tests.

Keywords: Support vector machines, unbalanced data sets, artificial points, support vectors, sensitivity, specificity, genetic algorithm.

1. Introducción

Un conjunto de datos esta no balanceado cuando contiene un gran número de patrones de muestra de un tipo (clase mayoritaria) y un número muy reducido de patrones de muestra opuestos a los anteriores (clase minoritaria). En el mundo existen aplicaciones que presentan un desbalance muy acentuado en su conjunto de entrenamiento, por ejemplo en problemas de detección de fraudes, donde el radio de desbalance puede ir de 100 a 1 hasta 100,000 a 1 [1], otros ejemplos son la clasificación de secuencias de proteína [2, 3], diagnóstico médico [4, 5], detección de intrusos y clasificación de texto [6, 7].

Experimentos recientes [8, 9, 10] han mostrado que el desempeño de la mayoría de los métodos de clasificación son afectados cuando son aplicados sobre conjuntos no balanceados, siendo más evidente cuando el radio de desbalance es muy alto, debido a que los clasificadores en general están diseñados para reducir el error promedio global sin importar la distribución de las clases.

Las Máquinas de Vectores Soporte (SVM) son actualmente una de las técnicas de clasificación más importantes [11, 3, 12, 13] debido a que tiene un mejor desempeño frente a otros métodos como redes neuronales artificiales [14, 15], árboles de decisión y clasificadores Bayesianos [2, 16]. Una SVM busca maximizar el margen de separación entre los híperplanos de cada clase, otorgándole un gran poder de generalización, propiedad que puede ser explicada por la teoría de aprendizaje estadístico [17]; sin embargo, en el caso de conjuntos no balanceados, el hiperplano de separación se ve sesgado hacia la clase mayoritaria, provocando un impacto negativo en la precisión de clasificación puesto que la clase minoritaria puede ser considerada como ruido y por consiguiente ignorada por el clasificador.

El desarrollo de nuevas técnicas para reforzar el desempeño de clasificadores como las SVM sobre conjuntos no balanceados es importante en el área de reconocimiento de patrones, minería de datos y aprendizaje de máquinas. Para mejorarlas surgen técnicas como bajo muestreo (*undersampling*) que balancea el conjunto al reducir la clase mayoritaria. Por su parte la técnica de sobre muestreo (*oversampling*) duplica la clase minoritaria tantas veces como sea necesario hasta equilibrar el tamaño de las clases [8]. La desventaja de estas técnicas es que al eliminar datos de forma aleatoria para la clase mayoritaria, se podrían estar eliminando datos importantes sobre la frontera de decisión, causando que el hiperplano de separación no sea el óptimo al usar una SVM; por el contrario, si se duplican los datos de la clase minoritaria, el tiempo de entrenamiento se incrementaría debido a que la complejidad de la SVM es $O(n^2)$ [2], además de que no aportan nada al entrenamiento puesto que estos puntos no son diferentes a los ya existentes.

Chawla et al. [18] propuso *Synthetic Minority Over sampling Technique* (SMOTE) que genera puntos sintéticos que son incluidos en la clase minoritaria. SMOTE toma un punto de la clase minoritaria y produce una nueva versión de este al desplazarlo hacia su vecino más cercano una distancia aleatoria para cada dimensión. Esta técnica

no incluye una selección de datos, opera con todo el conjunto de entrada y de acuerdo a los resultados es mejor que *undersampling* y *oversampling*.

Una combinación de SMOTE y oversampling fue propuesta en [8], introduciendo un esquema de penalización del error dependiendo de la clase, decrementando el costo para la clase mayoritaria e incrementándolo para la clase minoritaria; logrando una mayor densidad en la distribución de la clase minoritaria y colocando el hiperplano de separación más cerca de la clase mayoritaria. En [19] diferentes criterios de penalización son usados para producir efectos similares en la separación del hiperplano. Otras propuestas inspiradas en SMOTE pueden encontrarse en [20, 21, 22, 23]. Otro enfoque se basa en aplicar undersampling sobre conjuntos no balanceados seleccionando las muestras por medio de un algoritmo genético y resultando mejor que un simple muestreo aleatorio [24].

Para el caso de SVM con conjuntos no balanceados, en [25] logran mejorar su desempeño al generan datos sintéticos a partir de los vectores soporte (SV); mientras que en [26] estos SV son desplazados un épsilon. Ambos algoritmos trabajan en el espacio de características; sin embargo, sus desplazamientos son aleatorios obteniendo precisiones sesgadas hacia la sensitividad o la especificidad.

En este artículo se presenta una nueva técnica de muestreo a fin de mejorar el desempeño de las SVM sobre conjuntos no balanceados. Inicialmente se entrena la SVM usando el conjunto de entrenamiento completo con el objetivo de obtener los Vectores Soporte (SV) que serán usados como base para crear nuevos puntos artificiales y poblar la clase minoritaria. Este método incluye un algoritmo genético para encontrar una buena combinación entre parámetros de la SVM y puntos artificiales creados dentro de la clase minoritaria así como en la frontera entre clases.

En la metodología se explica cómo el método propuesto crea puntos artificiales de forma dirigida partiendo de los vectores soporte, que son los puntos más importantes, en lugar de usar todo el conjunto de datos de entrada como lo harían otras técnicas reportadas en la literatura; permitiendo crear puntos tanto dentro de la clase minoritaria como en la frontera entre clases. Para evitar introducir ruido con las nuevas muestras, el algoritmo es capaz de distinguir entre los SV más cercanos a la clase minoritaria. En la sección de resultados se puede observar que el método de clasificación propuesto mantiene el equilibrio entre sensitividad y especificidad al mismo tiempo que las mejora mientras que otras técnicas presentan sesgo hacia alguna de las dos métricas mencionadas.

2. Preliminares

2.1. Máquinas de vectores soporte (SVM)

Las SVM fueron inspiradas en los resultados de la teoría de aprendizaje estadístico desarrollado por Vapnik en los 70's [17]. Este clasificador permite encontrar un hiperplano capaz de separar linealmente dos clases, proyectando el espacio de entrada original a un espacio de características altamente dimensional donde maximiza el margen entre clases.

Las SVM permiten estimar una función de clasificación óptima empleando datos de entrenamiento etiquetados como X_{tr}, de esta forma, la función f clasificará

correctamente datos no vistos antes por el clasificador (datos de prueba). Considerando el caso más simple de clasificación binaria, asumimos que el conjunto X_{tr} es dado como:

$$(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n), \tag{1}$$

i.e. $X_{tr} = \{x_i, y_i\}_{i=1}^{n}$ donde $x_i \in R^d$ y $y_i \in R(+1, -1)$ corresponde a la etiqueta de clasificación de la muestra x_i. La función de clasificación puede ser escrita como:

$$y_i = sign\left(\sum_{i=1}^{n} \alpha_i y_i K\langle x_i \cdot x_j\rangle + b\right), \tag{2}$$

donde $x = [x_1, x_2, \dots, x_n]$ son los datos de entrada. Un nuevo objeto *x* puede ser clasificado usando (2). El vector x_i es mostrado en la forma de producto punto. Las α'_i son multiplicadores de Lagrange y b es el bias obtenido al entrenar la SVM.

2.2. Métricas para evaluar precisión en conjuntos no balanceados

Comúnmente, la precisión (*accuracy*) es la medida empleada para evaluar empíricamente el desempeño de un clasificador; sin embargo, para clasificación con datos no balanceados, esta métrica puede llevarnos a conclusiones erróneas debido a que la clase minoritaria tiene un impacto muy pequeño en la precisión. En conjuntos de datos con una distribución muy sesgada, la métrica de la precisión total no es suficiente debido a que en un conjunto descompensado de 99 a 1, un clasificador que etiquete todos los datos de prueba como negativos obtendrá una precisión del 99%, siendo inútil como clasificador para detectar los ejemplos positivos inusuales. La comunidad médica y la comunidad de aprendizaje de máquinas emplean dos métricas, la sensitividad (3) y la especificidad (4) para evaluar el desempeño de un clasificador sobre grandes conjuntos de datos altamente no balanceados.

$$S_n^{false} = \frac{T_N}{T_N + F_P}, \tag{3}$$

S_n^{true} es la proporción de verdaderos positivos i.e.,

$$S_n^{true} = \frac{T_P}{T_P + F_N}, \tag{4}$$

donde T_P es el número de patrones de clase +1 reales pronosticados como positivos (verdaderos positivos), T_N es el número de patrones de clase -1 reales pronosticadas como negativos (verdaderos negativos), F_P es el número de patrones de clase -1 reales pronosticados como positivos (falsos positivos) y F_N es el número de patrones de clase +1 reales que son pronosticados como negativos (falsos negativos).

2.3. Receiver operating characteristic (ROC)

La gráfica proporcionada por el método *Receiver Operating Characteristic* (ROC) es ampliamente usado para analizar el desempeño de clasificadores binarios y al medir

el área bajo la curva ROC (AUC) se obtiene una representación numérica de que tan separables son las clases analizadas [27]. Las ventajas más importantes del análisis con ROC es que no es necesario especificar los costos por errores de clasificación y proporciona una forma visual para analizar el desempeño del clasificador.

3. Metodología

3.1. Creación de puntos artificiales

El primer paso del método propuesto consiste en identificar las clases minoritaria y mayoritaria del conjunto de entrenamiento a partir del conjunto no balanceado original. La clase minoritaria contiene *t* muestras positivas etiquetadas como X_t^+, mientras que las muestras negativas X_t^- pertenecen a la clase mayoritaria. Si el conjunto negativo es muy grande, se aplica una técnica de bajo muestreo para evitar un alto costo computacional. El nuevo conjunto formado por X_t^+ y X_t^-posteriormente es empleado para entrenar una SVM, obteniendo un hiperplano preliminar $H_1(X_t^+, X_t^-)$ y su vectores soporte (SV).

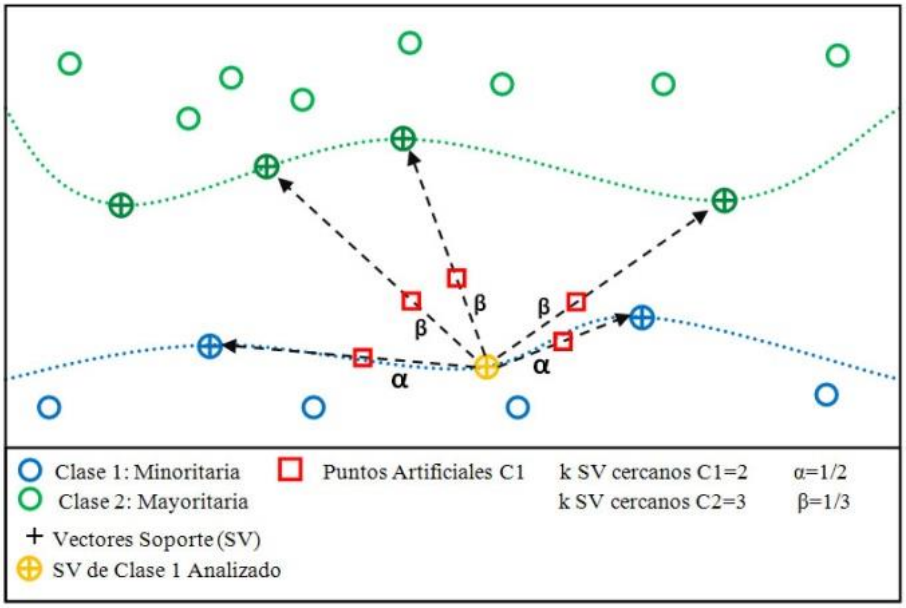


Fig. 1. Creación de puntos artificiales por el método propuesto

El segundo paso implica etiquetar los *SV* de acuerdo a la clase a la que pertenezcan, obteniendo SV^+ y SV^-; después son utilizados los SV^+ como referencia para crear nuevos *puntos artificiales* como se muestra en la Fig. 1, primero dentro de la clase minoritaria y después en su frontera con la clase mayoritaria. Para poblar la clase minoritaria es necesario elegir el número *k* de puntos que se crearán por cada SV^+ y el

desplazamiento α que moverá el SV_t^+ a una nueva posición. También hay que encontrar los $k\ SV^+$ más cercanos por cada SV^+ para finalmente aplicar (5).

$$X_{tk}'^{+} = SV_t^+ + \alpha * |SV_t^+ - SV_k^+| \quad \text{para cada } k\ SV_t^+ \text{ más cercano.} \tag{5}$$

Después de una forma similar, se poblará la frontera con puntos artificiales positivos pero ahora utilizando los $k\ SV^-$ más cercanos para cada SV^+; la proporción de desplazamiento β desplazará el SV_t^+ original en dirección de su vector soporte negativo más cercano de la siguiente forma:

$$X_{tk}''^{+} = SV_t^+ + \beta * |SV_t^+ - SV_k^-| \quad \text{para cada } k\ SV_t^- \text{ más cercano.} \tag{6}$$

La métrica usada para evaluar la distancia es la euclidiana y tanto α como β están en rangos entre 0 y 0.9, de esta forma se evita que un nuevo punto artificial pueda quedar localizado en el mismo lugar que el SV_k, ya que introduciría ruido.

3.2. Mejora de los parámetros usando un algoritmo genético

Con el fin de mejorar los parámetros de la SVM tales como tipo de *kernel*, costo *C*, *gamma* para el *kernel RBF*, así como los parámetros del método propuesto, $k\ SV^+$ más cercanos, $k\ SV^-$ más cercanos y los valores normalizados entre 0 y 0.9 para el desplazamiento α y β; se utilizó un algoritmo genético (GA) cuyo cromosoma contenía las variables anteriormente citadas. En la Tabla 1 se presenta el tamaño en bits ocupado por cada variable.

Tabla 1. Variables para el cromosoma

Variable	Rango	Precisión decimal	Tamaño en bits
gamma	[0.001, 1.000]	3	10
C	[0.001, 1.000]	3	10
Tipo de kernel	{1-lineal, 2-RBF}	0	1
k SV^+	{0,1,2}	0	2
α	[0.01, 0.90]	2	7
k SV^-	{0,1}	0	1
β	[0.01, 0.90]	2	7

El algoritmo genético encuentra una solución en un tiempo razonable y gracias a sus operadores de cruza y mutación puede realizar una búsqueda explotativa y explorativa respectivamente por lo que es mejor que utilizar una búsqueda por malla. Cada individuo dentro de la población tendrá un cromosoma como se ejemplifica en la Fig. 2 y su calidad como solución al problema estará en función de que tan próximas están las precisiones AUC, S_n^{true} y S_n^{false} respecto a la solución óptima que ocurre cuando todas valen 1.0. Para obtener los valores de cada métrica se requiere decodificar el cromosoma y obtener el fenotipo, asignando un valor a cada parámetro. Una vez que se tienen todos los valores se crean nuevos *puntos artificiales* y se entrena una SVM con los parámetros obtenidos.

El tamaño de cada gen se determinó con la fórmula (7), siendo *n* la precisión deseada.

$$nbits = log2[(limiteSuperior - limiteInferior)x10^n] + 0.5. \quad (7)$$

Para aplicar el mapeo de números binarios a reales se usó la fórmula (8), donde *nbits* es el tamaño de la palabra calculada anteriormente, *x'* es el valor obtenido de la conversión de la cadena binaria a decimal y *x* es el valor real obtenido de la transformación de *x'*.

$$x = limiteInferior + \frac{x'[limiteSuperior - limiteInferior]}{2^{nbits} - 1}. \quad (8)$$

3.3. Selección del modelo

El entrenamiento con SVM involucra el ajuste de varios parámetros que tienen un crucial efecto sobre el desempeño del clasificador entrenado. El modelo propuesto emplea una función de base radial (RBF) para entrenar la SVM, definida como:

$$K(x_i - x_j) = exp\left(-\gamma\|x_i - x_j\|^2\right), \gamma > 0. \quad (9)$$

El parámetro *C* regula el punto medio entre error de entrenamiento y complejidad, mientras que γ es un parámetro del *kernel*. La obtención de buenos parámetros se logró usando un algoritmo genético, fijando el tamaño del cromosoma en 38 bits, con una población de 24 individuos y seleccionando los padres en cada generación por sobrante estocástico con reemplazo. Los operadores usados son cruza de dos puntos y mutación uniforme con una probabilidad fija de 0.25, eligiendo el mejor individuo de dos corridas.

Para asegurar la convergencia, se aplicó el enfoque elitista, manteniendo intacto el material genético del mejor individuo en la siguiente generación y adicionalmente se utilizó una codificación Gray para disminuir las debilidades de la cruza de dos puntos.

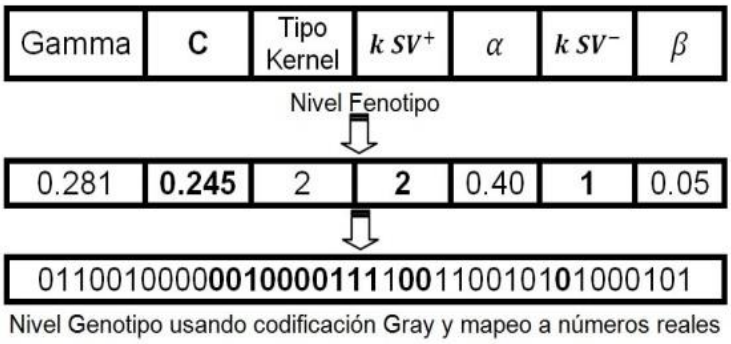
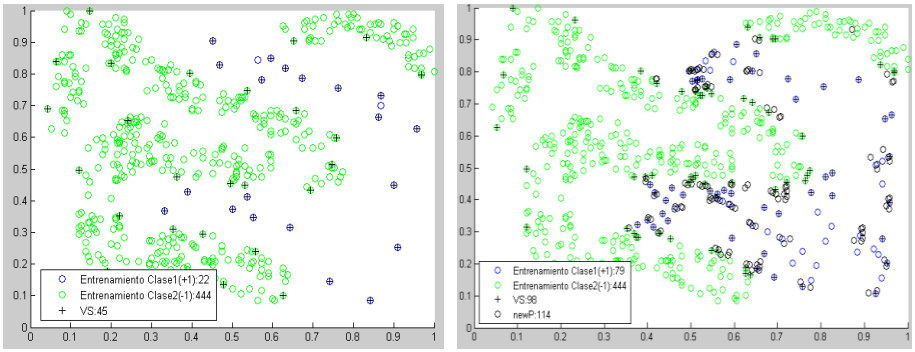


Fig. 2. Ejemplo de la estructura del cromosoma para el conjunto four class



a) Conjunto de entrenamiento no balanceado.

b) Conjunto de entrenamiento balanceado con puntos artificiales por el método propuesto.

Fig. 3. Distribución del conjunto four class, clase 1 en verde y clase 2 en azul

4. Resultados

En las pruebas realizadas se empleó el conjunto de datos KEEL Dataset, que es un conjunto comúnmente empleado para evaluar el desempeño de clasificadores sobre conjuntos de datos no balanceados; se encuentra disponible en http://sci2s.ugr.es/keel/datasets.php y los radios de desbalance van desde 1:1.4 para el conjunto *liver disorders* hasta 1:41.4 para *yeast 6*, que puede interpretarse como un patrón de muestra positivo por cada 41.4 patrones de muestra negativos correspondiente a este último conjunto.

Para realizar los experimentos se prepararon los conjuntos de entrenamiento y prueba, seleccionándolos aleatoriamente y destinando 80% y 20% respectivamente a partir del conjunto original. A continuación se presentan los resultados en dos secciones, primero usando las técnicas clásicas y después aplicando el método propuesto.

4.1. Desempeño de la SVM usando técnicas clásicas

En la Tabla 2 se presentan las precisiones AUC, S_n^{true} y S_n^{false}, donde cada valor corresponde al promedio de 10 pruebas para cada método, incluyendo la precisión alcanzada con el conjunto original. Puede notarse que las precisiones están sesgadas hacia S_n^{false} en el conjunto original, mientras que al aplicar *undersampling* u *oversampling* la precisión promedio no supera el 95% para las cuatro métricas y cuando alguna lo hace, la técnica sesga la precisión hacia una de ellas. En SMOTE, se usaron como parámetros un valor de 400 para *N* y 10 *k vecinos*. Esta técnica también tiene la debilidad de sesgar su precisión, clasificando la mayoría de patrones de prueba como positivos cuando está sesgada a S_n^{true} o clasificando la mayoría como negativos cuando está sesgada a S_n^{false}.

También se puede concluir que aunque el área bajo la curva ROC sea un valor alto para las técnicas analizadas, no puede usarse para discriminar si el clasificador es bueno detectando patrones positivos o si la técnica para balancear el conjunto es buena [27].

Tabla 2. Precisión para los conjuntos de datos no balanceados

Conjunto de datos	No balanceado			Undersampling			Oversampling			SMOTE		
	AUC	S_n^T	S_n^F	AUC	S_n^T	S_n^F	AUC	S_n^T	S_n^F	AUC	S_n^T	S_n^F
liver disorders	0.75	0.48	0.85	0.74	0.68	0.69	0.75	0.64	0.75	0.71	0.89	0.28
four class	0.87	0.51	0.97	0.87	0.78	0.78	0.88	0.81	0.80	0.83	0.91	0.71
glass 1	0.79	0.08	0.99	0.77	0.84	0.46	0.79	0.80	0.57	0.75	0.91	0.31
diabetes	0.81	0.56	0.86	0.81	0.74	0.71	0.81	0.69	0.76	0.79	0.83	0.62
glass 0	0.85	0.31	0.92	0.83	1.00	0.44	0.83	0.99	0.47	0.81	0.99	0.45
vehicle 2	0.99	0.90	0.98	0.99	0.96	0.91	0.99	0.82	0.98	0.99	0.97	0.93
vehicle 3	0.80	0.13	0.97	0.79	0.76	0.67	0.81	0.57	0.82	0.82	0.88	0.66
ecoli 1	0.95	0.69	0.96	0.94	0.92	0.84	0.94	0.89	0.86	0.95	0.91	0.84
ecoli 2	0.96	0.81	0.98	0.96	0.93	0.91	0.96	0.92	0.94	0.96	0.93	0.94
glass 6	0.93	0.70	0.98	0.96	0.80	0.95	0.94	0.80	0.98	0.96	0.80	0.98
yeast 3	0.98	0.66	0.98	0.98	0.91	0.93	0.97	0.65	0.98	0.98	0.86	0.96
ecoli 3	0.92	0.44	0.98	0.95	0.93	0.83	0.94	0.89	0.88	0.94	0.83	0.92
glass 2	0.66	0.00	1.00	0.64	0.97	0.31	0.64	0.87	0.37	0.64	0.00	1.00
cleveland 0 vs 4	0.98	0.15	1.00	0.94	0.95	0.70	0.97	0.20	0.99	0.98	0.60	0.99
glass 4	0.97	0.05	1.00	0.93	0.95	0.80	0.97	0.95	0.94	0.97	0.95	0.95
ecoli 4	1.00	0.75	1.00	1.00	1.00	0.93	0.99	0.90	0.98	1.00	0.93	0.99
page blocks 1-3 vs 4	1.00	0.50	1.00	0.99	0.98	0.90	1.00	0.90	0.98	1.00	0.94	1.00
glass 5	0.97	0.00	1.00	0.92	0.90	0.83	0.97	0.80	0.93	0.97	0.70	0.99
yeast 4	0.81	0.00	1.00	0.84	0.75	0.87	0.87	0.71	0.88	0.86	0.54	0.97
yeast 5	0.99	0.16	1.00	0.99	1.00	0.91	0.99	1.00	0.94	0.99	0.93	0.97
yeast 6	0.90	0.00	1.00	0.92	0.86	0.88	0.94	0.81	0.92	0.94	0.70	0.98

4.2. Desempeño de la SVM usando el método propuesto

En la Tabla 3 se presentan las precisiones AUC, S_n^{true} y S_n^{false} para el método propuesto con su respectiva desviación estándar, siendo cada valor el promedio de 10 pruebas.

Para balancear el conjunto de entrenamiento, en cada prueba se anexaron los puntos artificiales creados por el método propuesto y después para la prueba los parámetros *kernel RBF*, *C* y *gamma* requeridos por la SVM fueron calculados usando un algoritmo genético con codificación Gray.

Se observa que las precisiones mejoraron notablemente frente al conjunto original y no presentan gran sesgo, salvo para el conjunto *glass 0*, *glass 1* y *vehicle 3*; sin

embargo en comparación, estas precisiones son mejores frente a las otras técnicas analizadas.

Tabla 3. Precisión para los conjuntos de datos no balanceados usando el método propuesto

Conjunto de datos	Método propuesto (promedio)			Método propuesto (desviación estándar)			Puntos artificiales (clase positiva)
	AUC	S_n^T	S_n^F	AUC	S_n^T	S_n^F	
liver disorders	**0.93**	**0.89**	**0.81**	0.03	0.03	0.07	1326
four class	**1.00**	**1.00**	**1.00**	0.00	0.00	0.00	624
glass 1	**0.89**	**0.87**	**0.77**	0.04	0.05	0.04	54
diabetes	**0.87**	**0.85**	**0.80**	0.04	0.05	0.02	338
glass 0	**0.87**	**0.96**	**0.64**	0.04	0.05	0.05	65
vehicle 2	**1.00**	**1.00**	**0.98**	0.00	0.00	0.01	500
vehicle 3	**0.93**	**0.93**	**0.85**	0.02	0.02	0.03	668
ecoli 1	**0.96**	**0.95**	**0.89**	0.04	0.05	0.05	111
ecoli 2	**0.97**	**0.96**	**0.96**	0.04	0.05	0.03	48
glass 6	**0.98**	**0.90**	**0.99**	0.04	0.11	0.03	38
yeast 3	**0.98**	**0.98**	**0.96**	0.01	0.02	0.01	642
ecoli 3	**0.97**	**0.94**	**0.93**	0.02	0.07	0.03	135
glass 2	**0.99**	**1.00**	**0.97**	0.03	0.00	0.06	84
cleveland 0 vs 4	**1.00**	**1.00**	**1.00**	0.00	0.00	0.00	90
glass 4	**1.00**	**1.00**	**0.99**	0.00	0.00	0.01	88
ecoli 4	**1.00**	**1.00**	**0.99**	0.00	0.00	0.01	108
page blocks 1-3 vs 4	**1.00**	**1.00**	**1.00**	0.00	0.00	0.00	108
glass 5	**1.00**	**1.00**	**1.00**	0.00	0.00	0.00	24
yeast 4	**0.98**	**0.99**	**0.95**	0.02	0.03	0.04	62
yeast 5	**1.00**	**1.00**	**0.99**	0.00	0.00	0.01	238
yeast 6	**0.98**	**0.91**	**0.99**	0.04	0.07	0.01	84

Por último, la desviación estándar no superó el 0.04 para el AUC, el 0.07 para S_n^{true}, salvo *glass 6*, mientras que S_n^{false} tuvo un máximo de 0.07 por lo que sugiere que es un método estable.

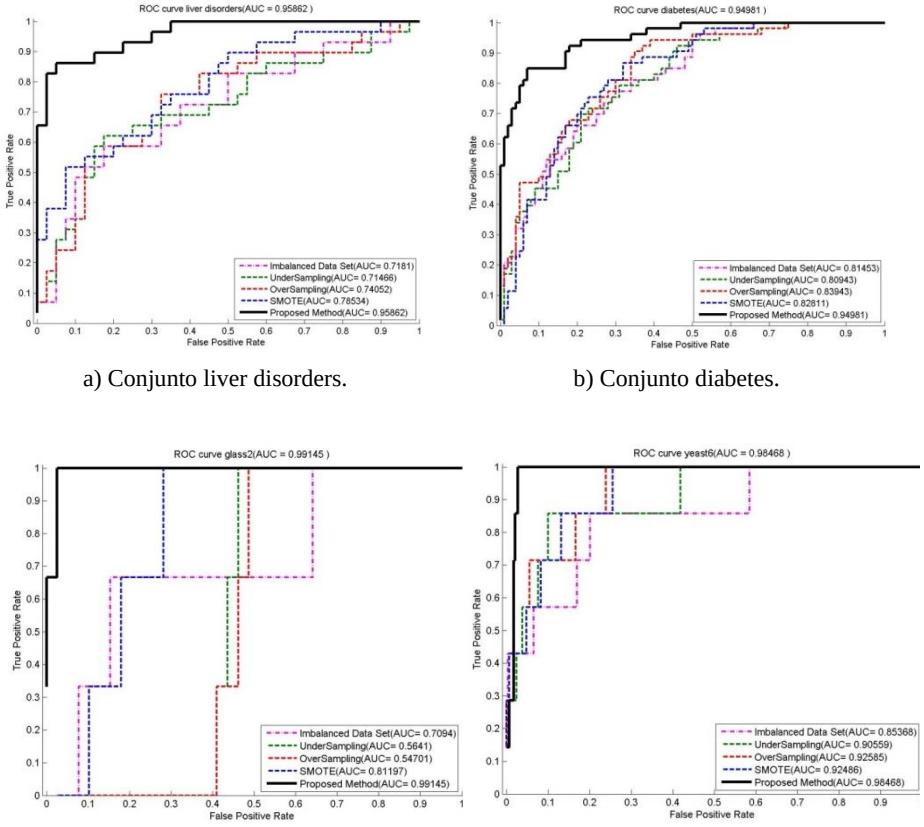
5. Conclusión

Las Máquinas de Vectores Soporte son una herramienta de clasificación que posee un buen desempeño sobre conjuntos balanceados; sin embargo, al trabajar en conjuntos desbalanceados, su desempeño es severamente afectado, ya que por la naturaleza de su entrenamiento el hiperplano obtenido se ve sesgado hacia la clase mayoritaria.

En este artículo, se presentó un nuevo método que mejora el desempeño de las SVM sobre conjuntos no balanceados, reduciendo el efecto del radio de desbalance al crear nuevos puntos artificiales que son agregados al conjunto de entrenamiento, al mismo

tiempo mejora significativamente el desempeño de las SVM en conjuntos con desbalance.

El método propuesto es diferente a otros métodos reportados en la literatura, ya que la creación de puntos es inteligente en el sentido de no utilizar como base todo el conjunto de datos de entrada, sino solo los Vectores Soporte, ya que son los puntos más importantes; sus dos fases pueden crear puntos tanto dentro de la clase minoritaria como en la frontera y al distinguir entre los SV más cercanos se puede evitar introducir ruido con los nuevos puntos.



a) Conjunto liver disorders. b) Conjunto diabetes.

c) Conjunto glass 2. d) Conjunto yeast 6.

Fig. 4. Gráficas ROC para una de las diez pruebas realizadas a los conjuntos a) liver disorders, b)diabetes, c) glass 2 y d) yeast 6

De acuerdo con los resultados, el método propuesto presentó una mejora en el desempeño de la SVM con precisiones superiores a las de las otras técnicas analizadas, disminuyendo el sesgo del hiperplano de separación al proveer el entrenamiento con más ejemplos de muestra positivos para la clase minoritaria y obteniendo resultados más notables cuando es aplicado sobre conjuntos de datos cuyo radio de desbalance es mayor a 10.

El método propuesto es estable, ya que su desviación estándar sobre la precisión no supera el 0.07, obteniendo una precisión S_n^{true} mayor o igual a 93% en 15 de los 22 conjuntos de datos no balanceados analizados y manteniendo una precisión alta para todas las precisiones, a diferencia de las demás técnicas, que presentan claramente una precisión sesgada hacia una de las métricas.

El desempeño en los conjuntos de datos probados es bueno; sin embargo, determinar la mejor combinación de parámetros, tanto para la SVM como para la creación de puntos artificiales es una tarea costosa computacionalmente, por lo que se incluyó un algoritmo genético dentro del algoritmo propuesto para obtener una buena combinación en un tiempo aceptable y mantener una buena precisión.

Referencias

1. Provost, F., Fawcett, T.: Robust Classification for Imprecise Environments. Machine Learning, pp. 203–231(2001)
2. Cervantes, J., Xiaoou, L., Wen, Y.: Splice site detection in DNA sequences using a fast classification algorithm. In: Proceedings of IEEE International Conference on System, Man and Cybernetics, pp. 2762–2767 (2009)
3. Dror, G., Sorek, R., Shamir, R.: Accurate identification of alternatively spliced exons using support vector machine. Bioinformatics, Vol. 21, No. 7, pp. 897–901 (2005)
4. Kononenko, I.: Machine learning for medical diagnosis: history, state of the art and perspective. Artificial Intelligence in Medicine, Vol. 23, pp. 89–109 (2001)
5. Grzymala-Busse, J.W., Stefanowski, J., Wilk, S.: A comparison of two approaches to data mining from imbalanced data. Journal of Intelligent Manufacturing, Vol. 16, pp.565–573 (2005)
6. Sebastiani, F.: Machine learning in automated text categorization. ACM Computing Surveys, Vol. 34, pp.1–47 (2002)
7. Tan, S.: Neighbor-weighted k-nearest neighbor for unbalanced text corpus. Expert Systems with Applications, Vol. 28, pp. 667–671 (2005)
8. Akbani, R., Kwek, S., Japkowicz, N.: Applying Support Vector Machines to Imbalanced Datasets. Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi, D. (Eds.): Machine Learning (ECML). Springer-Verlag Berlin Heidelberg, pp. 39–50 (2004)
9. Zeng, Z.Q., Gao, J.: Improving SVM Classification with Imbalance Data Set. In: Leung, C.S., Lee, M., Chan, J.H. (Eds.): Proceedings of the 16th International Conference on Neural Information Processing. Springer-Verlag, pp. 389–398 (2009)
10. Tezel, S.K., Latecki, L.J.: Improving SVM Classification on Imbalanced Data Sets in Distance Spaces. In: Ninth IEEE International Conference on Data Mining, pp. 259–267 (2009)
11. Bazzani, A., Bevilacqua, A., Bollini, D., Brancaccio, R., Campanini, R., Lanconelli, N., Riccardi, A., Romani, D., Zamboni, G.: Automatic detection of clustered microcalcifications in digital mammograms using an SVM classifier. In: Proceedings of European Symposium on Artificial Neural Networks, pp. 195–200 (2000)
12. Kong, W., Tham, L., Wong, K.Y., Tan, P.: Support Vector Machine Approach for Cancer Detection Using Amplified Fragment Length Polymorphism (AFLP) Screening Method. In: Proceedings of 2nd Asia-Pacific Bioinformatics Conference, Conferences in Research and Practice in Information Technology, pp. 63–66 (2004)
13. Platt, J.C.: Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines. Technical Report MSR-TR-98-14 (1998)
14. Arbach, L., Reinhardt, J.M., Bennett, D.L., Fallouh, G.: Mammographic Masses Classification: Comparison between Backpropagation Neural Network (BNN), K Nearest

Neighbors (KNN) and Human Readers. Electrical and Computer Engineering, Vol. 3, pp. 1441–1444 (2003)
15. Makal, S., Ozyilmaz, L., Palavaroglu, S.: Neural Network Based Determination of Splice Junctions by ROC Analysis. World Academy of Science, Engineering and Technology, No. 43, pp. 613–615 (2008)
16. Ya, Z., Chao-Hsien, Ch., Yixin, Ch., Hongyuan, Z., Xiang, J.: Splice site prediction using support vector machines with a Bayes kernel. Expert Systems with Applications, Vol. 30, No. 1, pp. 73–81 (2006)
17. Vapnik, V.N.: The Nature of Statistical Learning Theory. Springer-Verlag (1995)
18. Chawla, N., Bowyer, K., Hall, L., Kegelmeyer, W.: SMOTE: Synthetic Minority Over-sampling Technique. Journal of Artificial Intelligence Research, pp. 321–357 (2002)
19. Veropoulos, K., Campbell, C., Cristianini, N.: Controlling the Sensitivity of Support Vector Machines. In: Proceedings of the International Joint Conference on AI, pp. 55–60 (1999)
20. Hart, P.E.: The condensed nearest neighbor rule. IEEE Transactions on Information Theory, Vol. 14, pp. 515–516 (1968)
21. Han, H., Wang, W.-Y. , Mao, B.-H.: Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning. In: Huang, D.S., Zhang, X.-P., Huang, G.-B. (Eds.): Proceedings of the 1thInternational Conference on Intelligent Computing, Springer-Verlag, pp. 878–887 (2005)
22. Hu, S., Liang, Y., Ma, L., He, Y.: MSMOTE: Improving Classification Performance When Training Data is Imbalanced. In: Proceedings of the 2nd International Workshop on Computer Science and Engineering, Vol. 2, pp.13–17 (2009)
23. Hongyu, G., Herna, L.V.: Learning from imbalanced data sets with boosting and data generation: the DataBoost-IM approach. ACM SIG KDD Explorations Newsletter, Vol. 6, pp. 30–39 (2004)
24. Zou, S., Huang, Y., Wang, Y., Wang, J., Zhou, Ch.: SVM Learning from Imbalanced Data by GA Sampling for Protein Domain Prediction. In: Proceedings of The 9th International Conference for Young Computer Scientist, pp. 982–987 (2008)
25. Hernández, J., Cervantes, J., Trueba, A.: Mejorando la Clasificación de Datos No-Balanceados con SVM Generando Datos Sintéticos. Ciencia y Tecnología en Computación e Informática. CONACI, pp. 121–130 (2011)
26. Hernández, J., Cervantes, J., López, A., García, F.: Enhancing the Performance of SVM on Skewed Data Sets by Exciting Support Vectors. Pavón, J., Duque, N.D., Fuentes, R. (Eds.): Advances in Artificial Intelligence. Springer Berlin Heidelberg, pp. 101–110 (2012)
27. Fawcett, T.: An introduction to ROC analysis. Pattern Recognition Letters, Vol. 27, pp. 861–874 (2006)

Medidor inteligente para las variables de energía eléctrica basado en un sistema embebido

J. Álvarez-Alvarado[1], G.J. Ríos-Moreno[1], G. Ronquillo[2], M. Trejo-Perea[1]

[1] Universidad Autónoma de Querétaro, Facultad de ingeniería,
Santiago de Querétaro, Qro., México

[2] Centro de Ingeniería y Desarrollo Industrial (CIDESI),
Santiago de Querétaro, Qro., México

jose_manuel_51@hotmail.com, {riosg, mtp}@uaq.mx, gronquillo@gmail.com

Resumen. Hoy en día, cualquier sociedad o comunidad que deseen ser sostenible, debe pensar en la conservación y eficiencia de la energía eléctrica, que es un factor ético, donde se requiere el conocimiento del consumo total de energía con el fin de ayudar a preservar el medio ambiente. En este trabajo se presenta un sistema integrado basado en una FPGA para la medición y procesamiento de variables eléctricas y su interfaz para mostrar los datos procesados. Se muestran los resultados del trabajo y la comparación con un instrumento comercial de la marca Fluke 434.

Palabras clave: Sistema embebido, monitor de carga eléctrica, smart metter, FPGA, consumo de energía.

Smart Metter for Electrical Energy Variables Based on an Embedded System

Abstract. Nowadays, any society or community that wish to be sustainable, should think about conservation and energy efficiency, which is an ethical factor where require the knowing of the overall energy consumption in order to help preserve the environment. This paper presents an Embedded System based in an FPGA for the measurement and processing of electric variables, and its interface to show the data processed. The results of this system and the comparison with Fluke 434 are shown.

Keywords: Embedded system. electric load monitor. smart metter. FPGA. power consumption.

1. Introducción

El conocimiento de la demanda futura de energía eléctrica en una región, en un país o en el mundo, es una herramienta importante para el desarrollo e implementación de una política energética, ya sea por organizaciones internacionales o por el gobierno. Esta demanda de energía tendrá que ser satisfecha por una combinación óptima de las fuentes de energía disponibles, teniendo en cuenta las restricciones impuestas por el futuro cambio económico y social hacia un mundo sostenible [1, 2].

En los últimos años, el tema de la energía ha sido un tema abordado por los investigadores debido a su demanda en rápido crecimiento en todo el mundo. De acuerdo con la *International Energy Agency (IEA)*, el suministro de energía eléctrica podría ser un 50% mayor en 2030 de lo que hay en la actualidad y esto tendría consecuencias económicas y ambientales alarmantes [3, 4].

Actualmente, el consumo de electricidad en el mundo ha ido en aumento debido a varios factores, que han evolucionado con el tiempo, tanto tecnológico y social. Sin embargo, esto afecta afectando a la sostenibilidad del país, por lo que la necesidad de implementar un sistema de control para reducir este consumo es de gran importancia, ya que pueden alcanzarse no sólo el ahorro económico, sino que le ayudará a cuidar el medio ambiente.

La preocupación creciente entre las naciones industrializadas es el aumento del costo y el consumo de energía eléctrica. Al permitir a los consumidores controlar el consumo de energía de sus dispositivos eléctricos, pueden auxiliar a mejorar la conciencia y la disciplina de los usuarios de los hábitos domésticos [5].

Las computadoras, redes, la medición y el desarrollo de nuevas tecnologías han obtenido un impacto importante en el aspecto de la calidad de la energía. Los avances tecnológicos en redes, comunicaciones, gestión de datos y la tecnología aplicada a la medición de la calidad de energía se han combinado para reducir significativamente el costo de los sistemas de monitoreo y aumentar la capacidad de estos sistemas [6]. El procesamiento y la implementación de hardware de la matriz de puertas de campo programable (FPGA) en paralelo pueden reducir el tiempo total de cálculo para la interpretación de los datos y para ob-ner el estado de la red de distribución [7].

El uso de FPGA en monitoreo de sistemas eléctricos han sido usados en diversas investigaciones, como la propuesta de [8], el cual utiliza el FPGA para la adquisición y de datos para el consumo de energía en cargas individuales, utilizando convertidores análogos-digitales en paralelo, almacenando y enviando los datos a la PC.

El sistema de monitoreo propuesto por [9], realiza la adquisición de datos con un ADC0804, con una resolución de 8bits y almacenando los datos en una FIFO, obteniendo muestras de 60 muestras por ciclo.

El FPGA representa una buena opción para el desarrollo e implementación de monitoreo de energía debido a que permiten el desarrollo rápido de prototipos y para el diseño de complejos sistemas de hardware. Estos dispositivos se han utilizado en muchas aplicaciones reales, Las ventajas de la implementación de este sistema implican un bajo costo, una mayor precisión en la medición de variables eléctricas y un incremento de la operación flexibilidad al proporcionar monitoreo local y remoto a través de Internet [9].

La segunda sección de este documento se describe las consideraciones teóricas y los algoritmos propuestos para el cálculo y la programación de las fuentes de energía y parámetros en la FPGA. La tercera sección se encuentran los materiales y métodos aplicados, así como el desarrollo del hardware y software para el sistema propuesto e incluye la validación del sistema implantado y muestra la interfaz de realizada en LabView. La última sección presenta las conclusiones del sistema implementado y el impacto ecológico, social y económico de la investigación.

2. Consideraciones teóricas

A medida que la tecnología avanza, se espera que la complejidad de los equipos eléctricos aumenta al hacerlo también con el sistema eléctrico, de hecho, nos encontramos con tres tipos diferentes de cargas en edificios: puramente resistivas, puramente reactivas y parcialmente reactivos. Los instrumentos propuestos utilizan el procesamiento de señal digital para realizar un análisis espectral de la tensión y la corriente.

El método de integración discreta, calcula la corriente y voltajes señales y potencia activa a través de la versión digital de cada fuente de energía, en este orden, respectivamente. También permite el cálculo de los valores eficaces de la tensión, la corriente y, en consecuencia, la potencia aparente y el factor de potencia. Las muestras de corriente y de tensión que se ajustan, uno por uno, a los puntos idénticos en el tiempo, y el tiempo total de muestreo debe ser un número entero de *N* veces el período fundamental de las señales de entrada.

2.1. Medición de voltaje y corriente RMS

RMS se define para señales periódicas a pesar de que se utiliza generalmente para extraer información a partir de mediciones de perturbación sistema de energía que son no periódica. En caso de una transición, el RMS calculadas no da el valor correcto de Voltaje del nuevo estado hasta que la ventana sobre la que el RMS se calcula por completo contiene muestras del nuevo estado.

$$VRMS = \sqrt{\frac{1}{T}\int_{t_0}^{t_0+T}(v(t))^2\,dt} \cong \sqrt{\frac{1}{N}\sum_{n-1}^{N-1}(v(t))^2}\,, \tag{1}$$

$$IRMS = \sqrt{\frac{1}{T}\int_{t_0}^{t_0+T}(i(t))^2\,dt} \cong \sqrt{\frac{1}{N}\sum_{n-1}^{N-1}(i(t))^2}\,. \tag{2}$$

El valor RMS, que puede ser obtenido con métodos digitales, donde *N* es el total de las muestras de la señal adquirida digitalmente en observación, y *n* es un índice de la misma, *v* e *i* son los datos adquiridos por los sensores.

El valor RMS de voltaje en su forma discreta tiene las mismas propiedades: el valor eficaz que corresponde a una ventana que contiene las dos muestras de eventos previo

y posterior dará típicamente un valor de RMS que se encuentra entre la antigua y de la nueva tensión eficaz. Ciertas combinaciones de caída de magnitud y ángulo de fase de desempate incluso conducen a valores eficaces fuera de este rango [10].

2.2. Validación del sistema

Para calcular el porcentaje de error de los parámetros eléctricos de corriente, potencia activa tensión, potencia reactiva, potencia aparente y el factor, la siguiente ecuación se define de acuerdo a Cox [11].

$$\%error = 100\left(\frac{X_{Fluke} - Y_{system}}{Y_{system}} \right), \tag{3}$$

donde *Ysistem* indica la medida de los parámetros eléctricos por medio del Sistema desarrollado, *Fluke Y* representa los parámetros eléctricos medidos por el analizador de calidad de energía 43B.

3. Implementación del hardware

En éste proyecto, se propone utilizar un FPGA como una tarjeta de adquisición y procesamiento de datos, debido a que es de arquitectura abierta y a su flexibilidad, permite su reprogramación para diversas aplicaciones. Comparando el dispositivo comercial Fluke, el sistema propuesto es de bajo costo.

3.1. Características del FPGA

Para la adquisición y procesamiento de datos, fue seleccionada una FPGA Spartan-6 embebida en una tarjeta de Mojo [12]. La tarjeta mojo contiene la Spartan 6 XC6SLX9 FPGA, 84 pines IO digitales, 8 entradas analógicas, 8 focos LED de propósito general, en la regulación de voltaje de la tarjeta puede soportar de 4.8-12V, un ATmega32U4 utilizado para la configuración de la FPGA, comunicaciones USB, y la lectura de los pines analógicos y memoria flash *on-board* para almacenar el archivo de configuración de la FPGA.

Es posible realizar pruebas rápidas y obtener actualizaciones de las modificaciones de software individuales con el uso de FPGA y también tienen un coste de producción razonable en relación con su rendimiento.

3.2. Sensor de voltaje y corriente

El sensor de corriente utilizada en este proyecto es un núcleo dividido sensor de tipo CT SCT-013-000 (figura 1). El SCT-013 a 030 es un sensor de corriente AC no invasivo que puede detectar un valor de corriente máximo (*RMScurrent*) de 100A y el secundario del CT da salida a una tensión máxima de 1 V, con 2000 vueltas. Es un sensor salida

de tensión y por sí mismo viene con una resistencia de carga.[13]. Para determinar la resistencia de carga, se ha calculado el pico principal (*cc*) como 141,4A dado por:

$$pc = RMS_{current} * \sqrt{2} . \tag{4}$$

El pico de corriente secundaria es de 70,7mA, obtenido de la siguiente ecuación:

$$sc = \frac{pc}{No_{turns}} . \tag{5}$$

Para ello, la resistencia de carga se calcula mediante:

$$pc = \frac{2.5}{sc} . \tag{6}$$

El principio de funcionamiento de sensor de corriente y el sensor de SCT 013 se muestran en la Fig. 1.

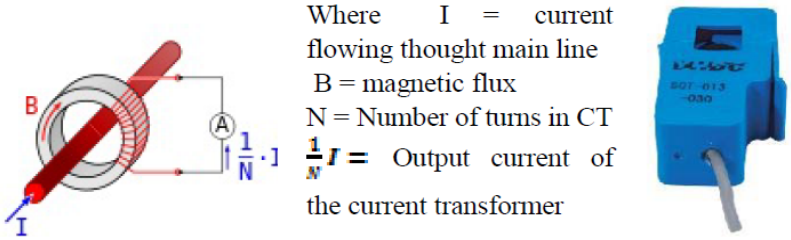


Fig. 1. Concepto básico del transformador de corriente y la imagen real del sensor de corriente (SCT-013-030) [13]

El sensor de tensión es un transformador de tensión, lo que asegura una red de energía protección galvánica. Este transformador se utiliza para convertir una tensión de 120 VAC en una tensión proporcional inferior. Ambas señales de corriente y tensión están montadas a un acondicionador de señal para aumentar el rango dinámico (± 5 V), para luego ser enviadas a un filtro pasa-bajas antes de completar la conversión de analógico a digital (A / D).

3.3. Interfaz digital

Para este proyecto, se utilizó el software LabVIEW, para hacer el análisis de las señales. Fig 2 muestra el circuito de acondicionamiento para montar la señal de Voltaje al ADS, se utiliza un amplificador operacional (LM358AD) para montar la señal de tensión en el rango positivo.

El sensor de corriente se conecta a un cable aux y en paralelo con el sensor de carga, para obtener la lectura de la señal en la interfaz de LabView..

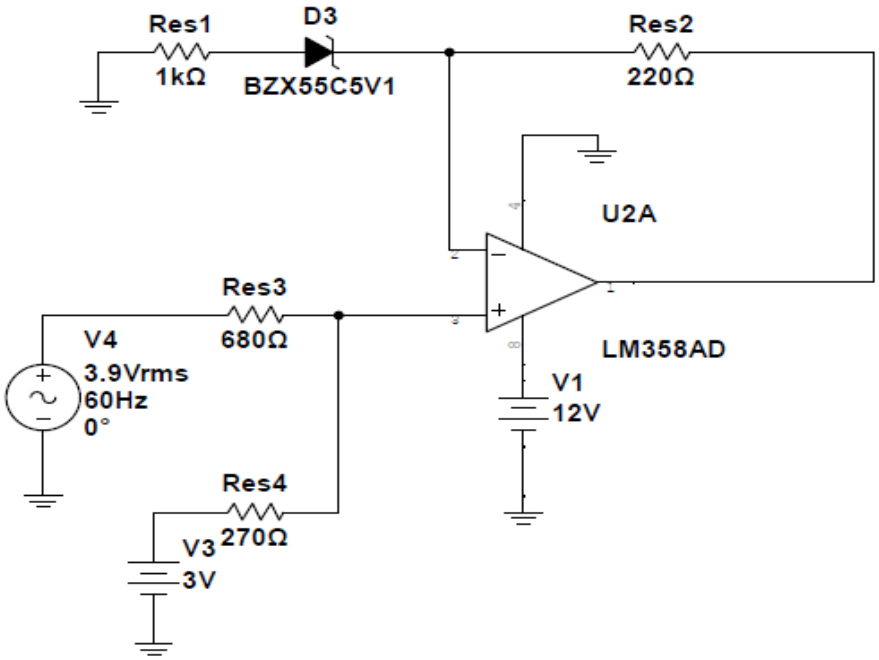


Fig. 2. Circuito de acondicionamiento de voltaje

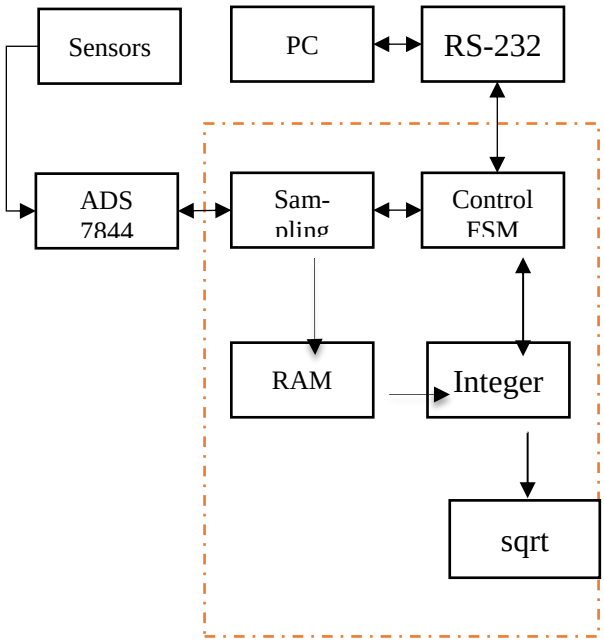


Fig. 3. Descripción general de la arquitectura

3.4. Implementación en hardware

Las señales de tensión y corriente pasan a través de un filtro antialiasing pasa-bajas analógico antes de completar la conversión de analógico a digital. Un DAQ realiza las conversiones entre las señales analógicas en señales digitales; un diagrama de bloques detallado de la DAQ se muestra en la Fig. 3. El convertidor analógico-digital seleccionado es un ADS7844 de 8 canales, 12 bits de resolución con una disipación de potencia de 3mW a una velocidad máxima de muestreo de 200 kHz y una alimentación de hasta +5V. El dispositivo que se utiliza para generar las señales de control ADC, establece la frecuencia de muestreo, almacena los datos, y la interfaz con el PC es el mojo FPGA Spartan-6. Como se muestra en la Fig. 3, los bloques funcionales se implementaron en el FPGA, basados en el trabajo de [9] y se explican posteriormente.

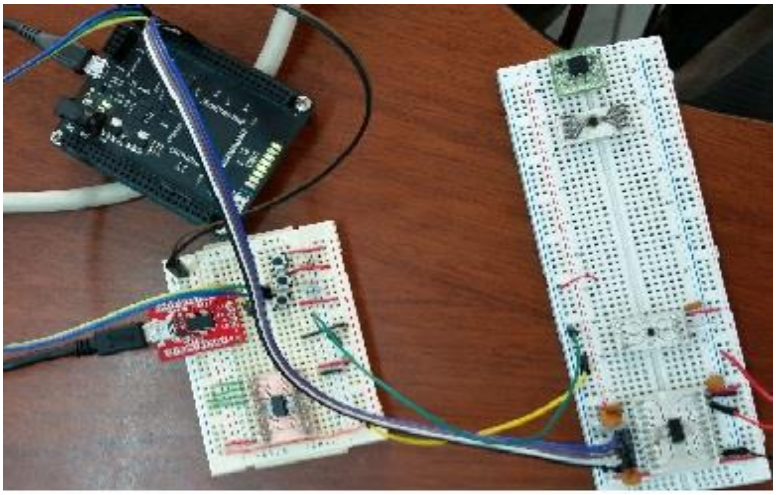


Fig. 4. Circuito de voltaje

- Control de la FSM: Este módulo es una máquina de estados finitos que genera las señales digitales requeridas por el ADC para iniciar un nuevo ciclo de conversión y almacenar los valores resultantes de estas conversiones en la memoria RAM, implementada en el FPGA.
- RAM: Cada valor convertido se almacena en el buffer de memoria y espera ser transmitido al equipo *host* a través del controlador periférico USB. Este módulo es una memoria RAM implementada y controlada por el FPGA.
- Base de tiempo: Este módulo establece la velocidad de muestreo de la conversión de analógico a digital. La frecuencia de muestreo se fijó en 1,2 kHz.

3.5. Simulaciones

En la Fig. 4. Muestra un circuito final, que muestra la tarjeta mojo, el circuito de acondicionamiento y el ADC para la adquisición de datos. Para hacer las medidas, era

necesario conocer el factor de potencia del circuito eléctrico. El banco de prueba se seleccionó una carga resistiva, para controlar la corriente y el voltaje.

La Fig. 5 muestra la adquisición de datos de tensión y la corriente eléctrica del sistema basado en FPGA y el analizador de calidad de energía eléctrica Fluke434 en una línea de tensión eléctrica en un banco de pruebas. Se realiza una comparación mostrando los valores reales del Fluke434 y los datos adquiridos en la interfaz gráfica por el sistema basado en FPGA que puede mostrar RMS de tensión y corriente eléctrica. El Fluke fue seleccionado para realizar la comparación entre los resultados obtenidos del mismo contra el sistema propuesto basado en FPGA, debido a que se ha utilizado en diversos trabajos como [9], [14].

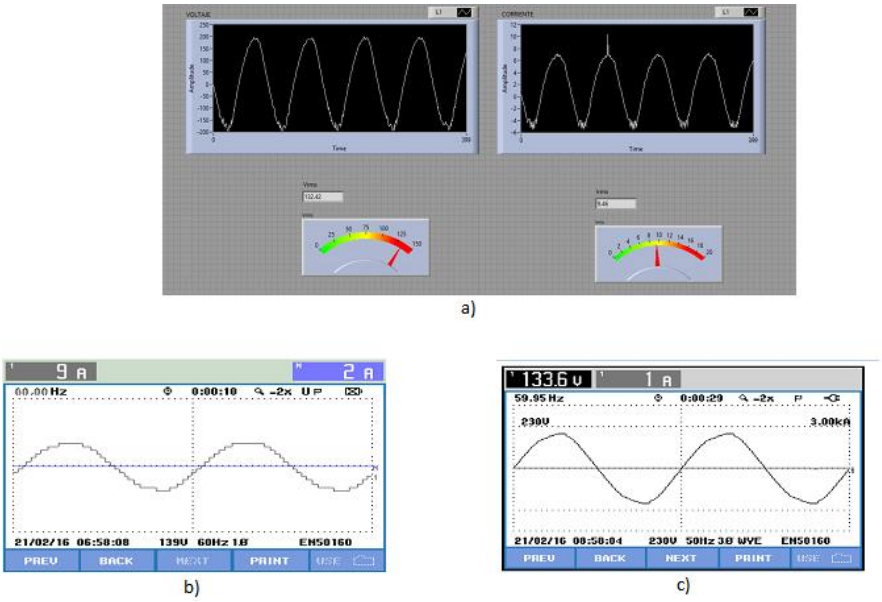


Fig. 5. Las mediciones de corriente y voltaje del sistema se muestran en a). En b) y c) se encuentran de las mediciones obtenidas con el analizador de calidad comercial Fluke para las variables de alimentación de corriente y tensión respectivamente

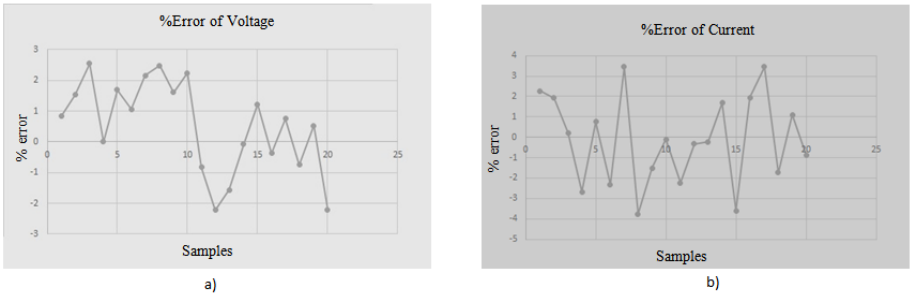


Fig. 6. Error measurements for voltage a) and current b)

Para la comparación y cálculo del error de la Ec. 3, se seleccionaron 20 mediciones basándose en [15] de acuerdo con el fin de ver los errores de tensión y corriente, y luego aplicar la ecuación (3). La Fig. 6 muestra el porcentaje de error del sistema.

4. Conclusiones y resultados

La principal característica del sistema de control propuesta es la flexibilidad y bajo costo. La flexibilidad garantiza que el software y el hardware puedan ser modificados y adaptados a diferentes mediciones o aplicaciones futuras, incluso para realizar estrategias de control. Así mismo, este sistema es comparativamente más fácil de calibrar y su reproducción digital es de mayor resolución, no causa problemas de linealidad que podría ocurrir en los contadores de energía basados en la reproducción analógica. El uso de sensores no invasivos permite la protección del sistema y para el usuario, si modificar la arquitectura eléctrica.

El ahorro sistema de monitoreo de energía eléctrica muestra una tasa de error por debajo de 3% y 4% de acuerdo a la Ec. 3 para las señales de tensión y corriente, respectivamente, que depende de la precisión de los convertidores analógicos digitales y el acondicionamiento de la señal analógica. Las fluctuaciones observadas pueden ser debidas a que se tomaron los valores comerciales, y no reales de la etapa de acondicionamiento de la señal.

Este sistema tiene como objetivo ser parte del diseño de un tablero eléctrico inteligente, con el fin de tener ahorros potenciales en la energía y económico, con la adición de un sistema experto como estrategia de control.

El sistema basado en FPGA propuesto resulta más económico y de una arquitectura abierta, por lo cual, podremos añadir diversas funciones al sistema propuesto en futuros trabajos de investigación.

Agradecimientos. Agradecemos especialmente al CONACYT (Consejo Nacional de Ciencia y Tecnología) por el apoyo económico y técnico, y a la Facultad de Ingeniería de la Universidad Autónoma de Querétaro.

Referencias

1. Morales-Acevedo, A.: ISES Solar World Congress Forecasting future energy demand : Electrical energy in Mexico as an example case. Energy Procedia. Vol. 57, pp. 782–790 (2014)
2. Álvarez-Alvarado, J., Trejo-perea, M., Herrera-Arellano, M., Ríos-moreno, J.G.: Control Strategies of Electrical Power on Smart Buildings , a Review. Int. Rev. Electr. Eng. Vol. 10, pp. 764–770 (2015)
3. Moslehi, K., Kumar, R.: Smart grid - A reliability perspective. In: Innov. Smart Grid Technol. Conf. ISGT, pp. 1–8 (2010)
4. Iea, I.E.A.: World Energy Outlook, http://link.aip.org/link/ELPWAQ/v23/i4/p329/s3&Agg=doi.

5. Marchman, C., Bertels, J., Gibbs, D., Students, S.N., Advisor, K.K.: Remote Monitoring and Control of Residential and Commercial Energy Use. In: Int. Telemetering Conf. Proc. pp. 1–8 (2014)
6. Trejo-Perea, M., Herrera-Ruiz, G., Rios-Moreno, J., Miranda, R.C., Araiza, E.R.: Greenhouse Energy Consumption Prediction using Neural Networks Models. Int. J. Agric. Biol. Vol. 11, pp. 1–6 (2009)
7. Depuru, S.S.S.R., Wang, L., Devabhaktuni, V.: Smart meters for power grid: Challenges, issues, advantages and status. Renew. Sustain. Energy Rev. Vol. 15, pp. 2736–2742 (2011)
8. Remscrim, Z., Paris, J., Leeb, S.B., Shaw, S.R., Neuman, S., Schantz, C., Muller, S., Page, S.: FPGA-based spectral envelope preprocessor for power monitoring and control. In: Conf. Proc. IEEE Appl. Power Electron. Conf. Expo. APEC, pp. 2194–2201 (2010)
9. Trejo-Perea, M., Herrera-Ruiz, G., Vargas-Vázquez, D., Luna-Rubio, R., Rios-Moreno, G.J.: System Electrical Power Monitoring Manifold Based on Software Development and an Embedded System for Intelligent Buildings. J. Energy, Vol. 137, pp. 1–10 (2011)
10. Styvaktakis, E.: Automatic classification of power system events using RMS voltage measurements. In: Power Eng. Soc. Summer Meet, pp. 824–829 (2002)
11. Cox, M.D., Williams, T.B.: Induction Varhour and Solid-State Varhour Meters Performances On Nonlinear Loads, Vol. 5, pp. 1678–1686 (1990)
12. Micro, E.: Mojo SPARTAN-6, https://embeddedmicro.com/tutorials/mojo.
13. Shajahan, A.H., Anand, A.: Data acquisition and control using Arduino-Android platform: Smart plug. Int. Conf. Energy Effic. Technol. Sustain. ICEETS, pp. 241–244 (2013)
14. Gonzalez-Gutierrez, C.A., Rodriguez-Resendiz, J., Mota-Valtierra, G., Rivas-Araiza, E.A., Mendiola-Santibañez, J.D., Luna-Rubio, R.: A PC-based architecture for parameter analysis of vector-controlled induction motor drive. Comput. Electr. Eng., Vol. 37, pp. 858–868 (2011)
15. Trejo Perea, M.: Monitoreo y ahorro de energía para edificios inteligentes. (2008)

Análisis de propiedades de medidas de similitud con atributos binarios

Iván Ramírez, Ildar Batyrshin

Instituto Politécnico Nacional, Centro de Investigación en Computación,
Ciudad de México, D.F.

{ramirez.alvarez.ipn, batyr1}@gmail.com

Resumen. El presente trabajo propone extender la visión basada en teoría de conjuntos sobre medidas de similitud en objetos con atributos binarios y que también pueden ser presentados en forma de tabla de contingencia. Se analiza las propiedades de diferentes medidas de similitud lo cual permite clasificarlas.

Palabras clave: Teoría de conjuntos, medidas de similitud, clasificación.

Analysis of Properties of Similarity Metrics with Binary Attributes

Abstract. We propose to extend an approach based on the set theory for similarity metrics with binary attributes that can be presented as a contingence table. We analyze different properties of similarity that allow us to classify them.

Keywords: Set theory, similarity metrics, classification.

1. Introducción

El análisis de medidas de similitud con atributos binarios y 2x2 tablas de contingencia ha recibido considerable atención durante muchos años en varios trabajos de minería de datos y reconocimiento de patrones [1,4-20]. Hasta este momento se han desarrollado decenas de medidas de similitud. Dependiendo del campo u objeto de estudio [6, 4], aplicación, estructura o la forma con que se trata a los atributos, se intenta seleccionar la medida más adecuada lo cual impactaría directamente en la precisión de los resultados obtenidos y además en la reducción de tiempo y proceso. Dicha tarea no es nada trivial puesto que en su estudio se consideran diferentes aspectos tales como cumplir con ciertas propiedades matemáticas [2] o que tan correlacionadas están dos medidas entre sí [5].

De manera informal una medida de similitud es una función cuyo valor real cuantifica la semejanza entre dos objetos. Esta es utilizada para medir hasta qué punto dos objetos, de acuerdo con los valores de sus atributos (características), son similares.

Es importante considerar los diferentes puntos de vista desde los cuales se analiza y mide la similitud entre instancias además de la forma utilizada por la medida en cuestión, por ejemplo medidas basadas en posibilidad y probabilidad [9, 21] que evalúan la generalidad y confiabilidad de ciertas reglas. Otra forma de medir similitud entre instancias es utilizando objetos cuyos atributos solo se encuentran en $\{0, 1\}$. Estos valores determinan la ausencia o presencia de una característica, pero más en general, una medida de similitud binaria estima las relaciones por las cuales dos objetos están siendo agrupados (relaciones taxonómicas).

Por otra parte en ocasiones también se consideran restricciones como "normalización" lo que obliga a que la medida se encuentre entre un intervalo dado, que por lo general se encuentra en [0, 1] o para ciertas medidas en [-1, 1] en donde inclusive en varios estudios no hacen explicita la diferencia de si una medida en cuestión se encuentra en los intervalos anteriores [7].

En este artículo son estudiadas diferentes propiedades que una medida de similitud deberá satisfacer. Se analizan aquellas que en la literatura se consideran fundamentales, tales como reflexividad y simetría; además de otras propuestas en [2, 3] como reflexividad estricta, débil similitud de reflexiones, cancelación de las reflexiones y no similitud de reflexiones.

Este trabajo propone una metodología para analizar y comparar diferentes medias de similitud basado en su cumplimiento de propiedades importantes para clasificar estas medidas y generar nuevas medias de similitud y de asociación con métodos considerados en [2,3]. Esta metodología está basada en la representación de medidas de similitud en términos de teoría de conjuntos que da un método sencillo de investigar propiedades de estas medidas. Se propone un agrupamiento de medidas basado en lo anterior. Las medidas utilizadas en este trabajo son más populares de área de reconocimiento de patrones pero son solo algunas de las mostradas en [7], sin embargo para trabajo futuro este estudio contempla ser extendido para todas estas medidas.

El resto de este trabajo está organizado de la siguiente manera: en la sección 2, se da la caracterización de las medidas de similitud utilizando conjuntos. En la sección 3 se realiza el estudio de las propiedades propuestas. En sección 4 se propone un agrupamiento basado en las propiedades estudiadas y por último en la sección 5 se exponen las conclusiones obtenidas.

2. Representación medidas de similitud en conjuntos

Sea $X = \{x_1, \dots, x_n\}$ el conjunto de atributos con valores binarios. Un objeto es un conjunto de atributos que este posee. Entonces sean $A \subseteq X$ y $B \subseteq X$ dos objetos que están definidos como conjuntos de atributos sobre X. Las medidas de similitud para datos binarios pueden ser expresadas como funciones de cuatro cantidades tales como lo presentado en Tabla 1:

a es número de atributos en común por los dos objetos A y B;

b es número de atributos en A pero no en B;

c es número de atributos en B pero no en A;

d es número de atributos que no se encuentran en ninguno de los dos A y B.

Tabla 1. Tabla de contingencia 2x2

	$\boldsymbol{B}$	$\bar{\boldsymbol{B}}$
A	a	b
$\bar{A}$	c	d

La misma tabla de contingencia aparece cuando A y B son variables binarias como $A = tiene\ gripa$ y B $= tiene\ temperatura\ alta$, en este caso el resultado de la muestra de $n = a + b + c + d$ personas está dado por:

a número de personas que poseen los atributos A y B;

b número de personas que poseen A pero no B;

c número de personas que poseen B pero no A;|

d número de personas que no poseen ninguno de los dos A y B.

En el siguiente texto para definir la caracterización de las medidas de similitud se utilizara la primera interpretación de Tabla 1. También esta tabla es posible escribirla en términos de conjuntos como en Tabla 2.

Tabla 2. Tabla de contingencia para los conjuntos A y B

	$\boldsymbol{B}$	$\bar{\boldsymbol{B}}$
A	$\lvert A \cap B \rvert$	$\lvert A \cap \bar{B} \rvert$
$\bar{A}$	$\lvert \bar{A} \cap B \rvert$	$\lvert \bar{A} \cap \bar{B} \rvert$

Lo anterior es importante debido a que existen diferentes caracterizaciones de esta tabla dependiendo del enfoque analizado [9, 15, 18]. Una vez establecido lo anterior se puede también reescribir aquellas medidas de similitud que trabajen con objetos binarizados. La Tabla 3 muestra las medidas más populares en las tareas de clasificación y reconocimiento de patrones para medir semejanza entre objetos cuyos valores de atributos son binarios con su correspondiente forma en conjuntos [7].

Como se sabe, un conjunto es una colección de objetos no ordenados, donde los elementos están separados por comas dentro de símbolos de llaves. No están ordenados debido a que $\{a, b\} = \{b, a\}$. Por lo tanto una medida de similitud $S(A, B)$ donde A y B son dos conjuntos también cumple con que: la similitud es grande cuando A y B están cerca, la similitud es pequeña cuando estos están lejos y normalmente es igual a 1 cuando estos son iguales (propiedad de reflexividad), y se encuentran entre el intervalo $[0, 1]$.

Lo anterior muestra que algunas medidas son idénticas entre sí a la hora de realizar la medición con sus valores, tal es el caso de las medidas de Dice y Czekanowski que tratan de la misma forma a sus variables. Además de Sokal y Michener se puede inferir que $|A \cap B| + |\bar{A} \cap \bar{B}| = \left|\overline{A \oplus B}\right|$, $|A \cap B| + |\bar{A} \cap B| + |A \cap \bar{B}| + |\bar{A} \cap \bar{B}| = |U|$ y $|A \cap B| + |\bar{A} \cap B| + |A \cap \bar{B}| = |A \cup B|$ para de esta manera aclarar su uso.

3. Propiedades estudiadas

Formalmente una medida de similitud está definida como una función $S: X \times X \rightarrow [0,1]$ que cumple con algunas propiedades consideradas posteriormente. En [2] y [3] se propone que estas deben cumplir también con las siguientes propiedades además de simetría y reflexividad:

P1. $S(A,B) = S(B,A)$	(simetría),
P2. $S(A,A) = 1$	(reflexividad),
P3. $S(A,B) < S(A,A)$ si $B \neq A$	(reflexividad estricta),
P4. $S(A,\bar{A}) < 1$	(débil similitud de reflexiones),
P5. $S(\bar{A},\bar{B}) = S(A,B)$	(cancelación de las reflexiones),
P6. $S(A,\bar{A}) = 0$	(no similitud de reflexiones).

Tabla 3. Definición de algunas medidas de similitud para datos binarios y su correspondiente representación en términos de conjuntos

	Representación Binaria	Representación en Conjuntos
1	$S_{\text{JACCARD}} = \frac{a}{a+b+c}$	$\frac{\lvert A \cap B \rvert}{\lvert A \cup B \rvert}$
2	$S_{\text{DICE,CZEKANOWSKI}} = S_{\text{NEI \& LI}} = \frac{2a}{2a+b+c}$	$\frac{2\lvert A \cap B \rvert}{2\lvert A \cap B \rvert + \lvert \bar{A} \cap B \rvert + \lvert A \cap \bar{B} \rvert}$
3	$S_{\text{3W-JACCARD}} = \frac{3a}{3a+b+c}$	$\frac{3\lvert A \cap B \rvert}{3\lvert A \cap B \rvert + \lvert \bar{A} \cap B \rvert + \lvert A \cap \bar{B} \rvert}$
4	$S_{\text{SOKAL \& SNEATH-I}} = \frac{a}{a+2b+2c}$	$\frac{\lvert A \cap B \rvert}{\lvert A \cap B \rvert + 2\lvert \bar{A} \cap B \rvert + 2\lvert A \cap \bar{B} \rvert}$
5	$S_{\text{SOKAL \& MICHENER}} = \frac{a+d}{a+b+c+d}$	$\frac{\lvert \overline{A \oplus B} \rvert}{\lvert U \rvert} = \frac{\lvert A \cap B \rvert + \lvert \bar{A} \cap \bar{B} \rvert}{\lvert U \rvert}$
6	$S_{\text{SOKAL \& SNEATH-II}} = S_{\text{GOWER \& LEGENDRE}} = \frac{2(a+d)}{2a+b+c+2d}$	$\frac{2(\lvert A \cap B \rvert + \lvert \bar{A} \cap \bar{B} \rvert)}{2\lvert A \cap B \rvert + \lvert \bar{A} \cap B \rvert + \lvert A \cap \bar{B} \rvert + 2\lvert \bar{A} \cap \bar{B} \rvert}$
7	$S_{\text{ROGER \& TANIMOTO}} = \frac{a+d}{a+2(b+c)+d}$	$\frac{\lvert A \cap B \rvert + \lvert \bar{A} \cap \bar{B} \rvert}{\lvert A \cap B \rvert + 2(\lvert \bar{A} \cap B \rvert + \lvert A \cap \bar{B} \rvert) + \lvert \bar{A} \cap \bar{B} \rvert}$

	Representación Binaria	Representación en Conjuntos
8	$S_{FAITH} = \frac{a + 0.5d}{a + b + c + d}$	$\frac{\lvert A \cap B \rvert + 0.5\lvert \bar{A} \cap \bar{B} \rvert}{\lvert U \rvert}$
9	$S_{RUSSEL\ \&\ RAO} = \frac{a}{a + b + c + d}$	$\frac{\lvert A \cap B \rvert}{\lvert U \rvert}$
10	$S_{OCHAI-I} = \frac{a}{\sqrt{(a + b)(a + c)}}$	$\frac{\lvert A \cap B \rvert}{\sqrt{(\lvert A \cap B \rvert + \lvert \bar{A} \cap B \rvert)(\lvert A \cap B \rvert + \lvert A \cap \bar{B} \rvert)}}$

En la tabla 4, se muestra la verificación de las propiedades P1 – P6 sobre las medidas propuestas dentro de la tabla 3, donde (●) implica que cumple con la propiedad y (○) que no la cumple.

Tabla 4. Tabla que muestra la verificación de las propiedades P1 – P6.

		P1	P2	P3	P4	P5	P6
1	$S_{JACCARD}$	●	●	●	●	○	●
2	S_{DICE}, $S_{CZEKANOWSKI}$, $S_{NEI\ \&\ LI}$	●	●	●	●	○	●
3	$S_{3W\text{-}JACCARD}$	●	●	●	●	○	●
4	$S_{SOKAL\ \&\ SNEATH\text{-}I}$	●	●	●	●	○	●
5	$S_{SOKAL\ \&\ MICHENER}$	●	●	●	●	●	●
6	$S_{SOKAL\ \&\ SNEATH\text{-}II}$, $S_{GOWER\ \&\ LEGENDRE}$	●	●	●	●	●	●
7	$S_{ROGER\ \&\ TANIMOTO}$	●	●	●	●	●	●
8	S_{FAITH}	●	○	●	●	○	●
9	$S_{RUSSEL\ \&\ RAO}$	●	○	●	●	○	●
10	$S_{OCHAI\text{-}I}$	●	●	●	●	○	●

Mostrar todas las comprobaciones de estas propiedades para cada una de las medidas de similitud tomaría mucho espacio, sin embargo se muestran algunas comprobaciones realizadas.

Para poder probar dichas propiedades, se retoman algunas identidades que resultan útiles, tales como:

$A - B = A \cap \bar{B}$ (diferencia entre conjuntos),
$A \cup A = A$, $A \cap A = A$ (leyes de idempotencia),
$\bar{\bar{A}} = A$ (ley de complementación),
$A \cup B = B \cup A$, $A \cap B = B \cap A$ (leyes de conmutatividad),

$\overline{A \cap B} = \bar{A} \cup \bar{B}$, $\overline{A \cup B} = \bar{A} \cap \bar{B}$ (leyes de De Morgan),
$A \cup \bar{A} = U$, $A \cap \bar{A} = \emptyset$ (leyes de complemento).

Ejemplo 1. La propiedad de simetría (P1) para Jaccard:

$$S(A,B) = \frac{|A \cap B|}{|A \cap B| + |\bar{A} \cap B| + |A \cap \bar{B}|}.$$

Se comprueba remplazando A por B respectivamente según lo establecido en la función S, dada por:

$$S(B,A) = \frac{|B \cap A|}{|B \cap A| + |\bar{B} \cap A| + |B \cap \bar{A}|}.$$

De lo anterior podemos observar que a través de las leyes de conmutatividad se obtiene:

$$S(A,B) = S(B,A)$$

es decir cumple con la propiedad.

Ejemplo 2. La propiedad de reflexividad (P2) para Faith:

$$S(A,B) = \frac{|A \cap B| + 0.5|\bar{A} \cap \bar{B}|}{|U|}.$$

Se comprueba de la siguiente forma:

$$S(A,A) = \frac{|A \cap A| + 0.5|\bar{A} \cap \bar{A}|}{|U|}.$$

Utilizando las identidades de leyes de complementos y de idempotencia es posible ver que $|A \cap A| = |A|$, $|\bar{A} \cap \bar{A}| = |\bar{A}|$ y $|\bar{A} \cap A| = 0$, $|A \cap \bar{A}| = 0$ respectivamente para los términos $|\bar{A} \cap B|$ y $|A \cap \bar{B}|$ en $|U|$. Por lo tanto se obtiene:

$$S(A,A) = \frac{|A| + 0.5|\bar{A}|}{|A| + |\bar{A}|}.$$

De ahí que para esta medida se tiene:

$$S(A,A) \neq 1,$$

es decir no cumple con la propiedad.

Ejemplo 3. La propiedad de no similitud de reflexiones (P6) para Russel & Rao:

$$S(A,\bar{A}) = \frac{|A \cap \bar{A}|}{|U|}.$$

Inmediatamente por ley de complemento se tiene que $|\mathrm{A} \cap \overline{\mathrm{A}}| = |\emptyset| = 0$ y por lo tanto:

$$S(A, \bar{A}) = 0,$$

es decir, cumple con la propiedad.

4. Agrupamiento por propiedades

Basándose en la tabla 4 entonces se puede agrupar en diferentes clases a estas medidas. Dichas clases pueden ser parcialmente ordenadas de tal manera que una clase contiene a otra. Las clases quedan definidas por las clases $C1, C2$ y $C3$ respectivamente y cada una de ellas está dada por:

- $C1 = \{P1 - P6\}$ y contiene a las medidas Sokal & Michener, Sokal & Sneath-II, Gower & Legendre y Roger & Tanimoto, con números 5 – 7.
- $C2 = \{P1 - P4, P6\}$ y contiene a las medidas Jaccard, Dice, Czekanowski, Nei & Li, 3W-Jaccard, Sokal & Sneath-I y Ochai-I o basándose en la numeración de la tabla 4, con números 1 – 4 y 10.
- $C3 = \{P1, P3, P4, P6\}$ y contiene a las medidas Faith y Russel & Rao o las con números 8 y 9.

Lo anterior también puede verse utilizando una notación conjuntista de inclusión donde:

$$C3 \subseteq C2 \subseteq C1.$$

5. Conclusiones

Este artículo presenta un estudio en relación a medidas de similitud populares en diferentes áreas como clasificación y reconocimiento de patrones, utilizando y extendiendo el punto de vista de conjuntos lo que permite analizar a estas medidas basándose en las propiedades propuestas P1 – P6. Este análisis permite ver que estas medidas pueden ser agrupadas en diferentes clases $C1, C2, C3$. Se puede mencionar que se encontró que una medida bastante popular como Russel & Rao que pertenece a clase $C3$ no cumple con propiedad de reflexividad P2 que normalmente está considerada como obligatoria para medidas de similitud; por esto el uso de esta medida en tareas de clasificación y reconocimiento de patrones es cuestionable. Se debe mencionar que las medidas de la clase $C1$ en comparación con las de la clase $C2$ cumplen una importante propiedad como lo es P5 y debido a esto resultarían más útiles en algunas tareas de análisis de datos.

Para trabajo futuro, a partir de las propiedades estudiadas puede extenderse continuando con el análisis sobre la mayoría de medidas de similitud, disimilitud y asociación conocidas en varios estudios [7, 9, 18] para establecer sus propiedades y clasificarlas dependiendo de lo anterior y generar nuevas medidas que tienen propiedades deseables.

Agradecimientos. El presente trabajo fue apoyado por el proyecto SIP 20162204 y BEIFI del IPN.

Referencias

1. Batagelj, V., Bren M.: Comparing resemblance measures. Journal of classification, Vol. 12, No. 1, pp. 73–90 (1995)
2. Batyrshin, I.Z.: Association measures on [0, 1]. Journal of Intelligent and Fuzzy Systems, Vol. 29, No. 3, pp. 1011–1020 (2015)
3. Batyrshin, I.Z.: On definition and construction of association measures. Journal of Intelligent & Fuzzy Systems, Vol. 29, No. 6, pp. 2319–2326 (2015)
4. Cha, S.H., Srihari, S.N.: On measuring the distance between histograms. Pattern Recognition, Vol. 35, No. 6, pp. 1355–1370 (2002)
5. Cha, S.H.: Comprehensive survey on distance/similarity measures between probability density functions. City, Vol. 1, No. 2, p. 1 (2007)
6. Cha, S.H.: Taxonomy of nominal type histogram distance measures. City, Vol. 1, No. 2, p. 1 (2008)
7. Choi, S.S., Cha, S.H., Tappert, C.C.: A survey of binary similarity and distance measures. Journal of Systemics. Cybernetics and Informatics, Vol. 8, No. 1, pp. 43–48 (2010)
8. Dunn, G., Everitt, B.S.: An introduction to mathematical taxonomy. Courier Corporation, (2004)
9. Geng, L., Hamilton, H.J.: Interestingness measures for data mining: A survey. ACM Computing Surveys (CSUR), Vol. 38, No. 3, p. 9 (2006)
10. Gower, J.C., Legendre, P.: Metric and Euclidean properties of dissimilarity coefficients. Journal of classification, Vol. 3, No. 1, pp. 5–48 (1986)
11. Guillet, F., Hamilton, H.J. (Eds.): Quality measures in data mining (43). Springer, (2007)
12. Hinkin, T.R.: A brief tutorial on the development of measures for use in survey questionnaires. Organizational research methods, Vol. 1 No. 1, pp. 104–121 (1998)
13. Jalali-Heravi, M., Zaïane, O.R.: A study on interestingness measures for associative classifiers. ACM Symposium on Applied Computing, pp. 1039–1046 (2010)
14. Lerman, I.C.: Les bases de la classification automatique. Gauthier-Villars, (1970).
15. Lesot, M.J., Rifqi, M., Benhadda, H.: Similarity measures for binary and numerical data: a survey. International Journal of Knowledge Engineering and Soft Data Paradigms, Vol. 1, No. 1, pp. 63–84 (2008)
16. Li, Y., Qin, K., He, X.: Some new approaches to constructing similarity measures. Fuzzy Sets and Systems, Vol. 234, pp. 46–60 (2014)
17. Tan, P.N., Steinbach, M., Kumar, V.: Introduction to data mining (1). Boston: Pearson Addison Wesley (2006)
18. Tan, P., Kumar, V., Srivastava, J.: Selecting the Right Interestingness Measure for Association Patterns. ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining (2002)
19. Veal, L.R.: Syntactic Measures and Rated Quality in the Writing of Young Children. Studies in Language Education, Vol. 8 (1974)
20. Vo, B., Le, B.: Interestingness measures for association rules: Combination between lattice and hash tables. Expert Systems with Applications, Vol. 38, No. 9, pp. 11630–11640 (2011)
21. Zadeh, L.A.: A note on similarity-based definitions of possibility and probability. Information Sciences, Vol. 267, pp. 334–336 (2014)

Evolución de descriptores estadísticos de superficie de imágenes por programación genética para el reconocimiento de imágenes por CBIR: una primera aproximación

Héctor Alejandro Tovar Ortiz[1], César Augusto Puente Montejano[1], Juan Villegas-Cortez[2], Carlos Avilés Cruz[2]

[1] Universidad Autónoma de San Luis Potosí, Facultad de Ingeniería, San Luis Potosí, SLP, México

[2] Universidad Autónoma Metropolitana, Azcapotzalco, Departamento de Electrónica, Cuidad de México, México

tovarha12@gmail.com, cesar.puente@uaslp.mx, {juanvc, caviles}@azc.uam.mx

Resumen. El crecimiento sostenido de la Internet en nuestra sociedad se vuelve más evidente al considerar que más de la mitad de la información en ella son imágenes, y el gran problema por atacar sigue siendo el reconocimiento de imágenes de forma automática y no supervisada. La metodología de reconocimiento de imágenes a partir de su contenido propio (CBIR) ha mostrado tener mucha efectividad, tratando de simular el comportamiento de clasificación natural de las imágenes como los seres humanos lo realizamos: verla, analizarla y determinar su clasificación. En este trabajo presentamos la solución de crear nuevos descriptores estadísticos que apoyen la metodología CBIR, aplicados sobre imágenes de escenarios naturales, por medio de la Programación Genética, a partir de un conjunto de funciones intuitivo, y con el alto costo computacional de considerar a la función de aptitud como un clasificador de distancia mínima y auto organización de racimos de atributos, logrando hasta este momento, una clasificación y reconocimiento de casi el 100 %.

Palabras clave: Algoritmos evolutivos, programación genética, reconocimiento de imágenes, CBIR, reconocimiento de patrones.

CBIR Pattern Recognition Using Genetic Programming: First Approximation Using New Descriptors for Images Textures

Abstract. The steady increase of Internet in our society becomes more evident against the crude fact that more than a half of the internet infor-

mation consists of images. This situation poses an open problem, how to recognize images automatically and unsupervised. The methodology for recognizing images by means of their own content (CBIR) has shown a lot of effectivity. CBIR resembles the orderly way in which humans manage to recognize images: seeing, analyzing and then classify them. In this work we present a solution to this problem by creating new statistical descriptors supporting the CBIR methodology. These are applied over images of natural scenes by means of Genetic Programming using a set of intuitive functions. This process involves the use of an amplitude function as a classifier of the self organization and the minimum distance of the attribute clusters. The incorporation of this last feature comes with a high computational cost. However, up to date, our results get a classification and recognition of almost 100 %.

Keywords: Evolutionary algorithms, genetic programming, image recognition, CBIR, pattern recognition.

1. Introducción

En los últimos años el crecimiento sostenido de la Internet ha impactado diferentes áreas del crecimiento humano, entre ellos esta el conocimiento compartido por medio de la información que en ella se dispone, y acorde a [6] más de la mitad de ésta es en imágenes. El problema de manejar imágenes en grandes cantidades, como a las que hemos llegado con el Big Data, es que éstas no están nombradas acorde a su contenido, y por un lado se requiere poder apoyarse en las imágenes para diferentes propósitos de discurso, y por el otro las mismas imágenes en cantidad van creciendo; por ello se sigue trabajando en la comunidad mundial del área científica en poder desarrollar nuevas formas de su almacenamiento, distribución, alta disponibilidad y su clasificación acorde a diferentes propósitos. Los seres humanos podemos de forma simple clasificar un conjunto de imágenes, es algo que hacemos cotidianamente, pero lograr que una computadora lo haga no es tarea sencilla, y en esta tarea se centra el trabajo aquí presentado, en cómo mejorar una de las técnicas de clasificación automática de imágenes, que parte del contenido propio de cada imagen obtenido tras su análisis de superficie, llamada CBIR (Content Based Image Retrieval) [6].

El Cómputo Evolutivo ha mostrado brindar nuevas soluciones para problemas que han sido resueltos de forma determinísitica, e.g., los problemas de programación no lineal, o de optimización; específicamente en este trabajo hemos aplicado la metodología evolutiva de la Programación Genética, como herramienta fundamental que nos permite la exploración de nuevos programas-individuos de una población, y su desempeño como posible solución al problema formulado por cada uno de los programas evolucionados. Es así, que con la metodología planteada se ha logrado, hasta este momento, hallar nuevas soluciones que resuelven un problema muy complejo, haciendo variaciones sobre las imágenes usadas para el entrenamiento de un clasificador, brindando soluciones factibles.

En la Sección 2 se muestra el estado del arte de nuestro alcance, en la Sección 3 compartimos la metodología planteada de CBIR+PG, en la Sección 4 presen-

tamos la experimentación y los resultados obtenidos, y finalmente compartimos las conclusiones alcanzadas en la Sección 5.

2. Estado del arte

La disciplina de la recuperación de imágenes a partir del contenido de las mismas, o CBIR, es una de las más activas, considerando que ha ido madurando desde la fecha clave de febrero de 1992, donde la US National Science Fundation (USNSF) organizara una primera reunión de trabajo en Redwood, California, para identificar las áreas de investigación principales que deberían considerarse en cuenta por los investigadores para la administración de los sistemas de información visual, mismo que pudieran ser de utilidad para la ciencia, industria, medicina, medio ambiente, educación, entretenimiento y otras aplicaciones [9]; posteriormente a ésta reunión el navegador de Internet Mosaic fue liberado, iniciando una revolución en la web que iría cambiando rápidamente, y en ese tiempo también se fueron desarrollando nuevos foto sensores y haciéndose disponibles; con todo esto, el número de imágenes almacenadas comenzó a incrementarse y la necesidad de crear herramientas que ayudaran a su ordenamiento, clasificación e indexación fue algo prioritario a desarrollar [8].

Usando CBIR se ha tratado de poner un orden las imágenes, a fin de poder lograr su clasificación para una rápida identificación. Actualmente se han desarrollado aplicaciones con un alto nivel de reconocimiento tanto para imágenes en escenarios naturales [6], así como para identificar rostros humanos [1], con la particularidad en ésta última aplicación a rostros de que se considera la información de la "textura" del rostro de la persona, y a partir de éstas características se realiza la clasificación. Una metodología general de CBIR se muestra a la derecha de la figura 1, en ese se aprecia cómo a partir de un conjunto de imágenes de entrenamiento se lee cada imagen, se cambia al espacio de color HSI, donde se tiene más información no perceptible a nuestros ojos (que sólo vemos en el espacio de color RGB), luego sobre la nueva imagen se realiza un sembrado de puntos aleatorios, y en cada punto se abre una ventana de tamaño 10×10 pixeles, y sobre ésta ventana se extraen una tripleta de características estadísticas. En [6] y [1] se muestra el uso de tres descriptores de superficie: la media, la desviación estándar y la homogeneidad (obtenida a partir de los valores de la matriz de co-ocurrencia) [3]; y es con esta tripleta de valores que se logra hacer la caracterización de las imágenes a partir de un análisis local, extrapolándose hacia toda la imagen.

Los Algoritmo Evolutivos (AE) han mostrado ser de utilidad en brindar soluciones factibles, para problemas debidamente formulados con restricciones [2]; y hablando sólo de la posibilidad de acoplar un algoritmo genético (AG) para una mejor disposición del número de racimos dispuestos para aglomerar los patrones obtenidos se tiene en [5]. Como parte de los AE tenemos a la Programación Genética (PG), que a diferencia de loa AG, no se limitan en su búsqueda evolutiva a una cadena genómica de longitud finita, como se formulan inicialmente los AG; la PG se basa en la evolución de programas completos, don-

de cada uno es una solución candidata al problema, siendo ésta su formulación fundamental dada por J. Koza [4].

En este trabajo proponemos la aplicación de la coevolución, desde el principio de "divide y vencerás" de la formulación del modelo del problema a resolver, donde cada población depende de la otra, o se dice que tiene una relación estrecha. Los modelos de coevolución básicos son: cooperativa y presa-cazador, el mejor ejemplo de la primera son las colonias de hormigas, donde entre todos los individuos de la población buscan cooperar para hallar una solución al problema, mientras que en el segundo modelo, una parte es la que busca sobrevivir, la presa o el cazador, pero si una de las dos especies muere, la otra también lo hará, de ahi que se tiene que llegar a un equilibrio entre las especies. Dado lo anterior, en este trabajo formulamos una coevolución cooperativa, inspirada en el modelo de Memorias Asociativas Evolutivas (MAE) [10], con la finalidad de proponer nuevos descriptores locales de superficies de imágenes para ser usados desde la metodología CBIR.

3. Metodología

Nuestra solución propuesta es la combinación de CBIR con PG para identificar imágenes de escenarios naturales, y para esta primera aproximación usamos la base de imágenes de J. Voguel [11], con la metodología CBIR reportada en [6] y mejorada en [1], misma que se detalla en general sobre el lado derecho de la figura 1. La aplicación de PG se realiza en evolucionar el conjunto de descriptores como una tripleta, cada individuo entonces se coevoluciona en dos fases, que son detalladas más abajo; siendo en la primera la evolución de una pareja de descriptores, y los ganadores de al menos N procesos se aportarán en el segundo proceso o fase, para lograr el tercer descriptor, repitiéndose N veces, para al final de todo el proceso tener una tripleta de descriptores de la ventana de superficie de la imagen a estudiar.

El algoritmo se divide en dos fases (procesos evolutivos 1 y 2), que se explican a continuación. Durante el primer proceso evolutivo se buscan los primeros dos elementos de la tripleta final, mediante un algoritmo evolutivo de PG. En el segundo proceso evolutivo, se toman los pares generados en la fase anterior y se utilizan para sintetizar tripletas mediante otro algoritmo evolutivo de PG. Los parámetros de la PG considerados como fundamentales para los dos procesos evolutivos considerados son: el conjunto de funciones F_a , el conjunto de terminales T_a y la función de aptitud F, tal como se describen a continuación:

- $F_a := \{+, -, *, mydivide, sen, cos\}$,
- T_a : {El conjunto de descriptores estadísticos: $fd01, fd02, \ldots, fd09$},
- F : la función de aptitud : el porcentaje de recuperación sobre toda la base de datos (DB) considerada, entrenando el clasificador sobre CBIR con los descriptores evolucionados, éste valor es usando la técnica de resustitución, i.e., se entrena y prueba el clasificador usando toda la DB. Aquí consideramos el óptimo alcanzar el 100 %, acorde con la ecuación 1, tal que $f \in [0, 1]$, siendo el mejor valor 1.

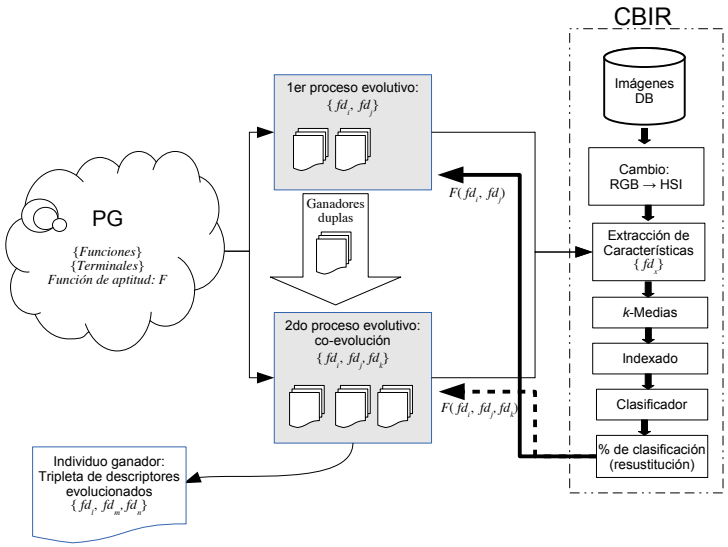


Fig. 1. Metodología general propuesta de evolución de descriptores de textura de imágenes, en coevolución cooperativa, usando CBIR

$$F = \frac{Num \quad de \quad imagenes \quad recuperadas}{Num \quad de \quad imagenes \quad totales}. \tag{1}$$

Los descriptores considerados se calculan sobre cada ventana por capa, para esta primera aproximación evolutiva que estamos reportando, y son los siguientes:

- $fd01 : fd1_H =$ media,
- $fd02 : fd2_H =$ desv. std,
- $fd03 : fd3_H =$ homogeneidad-coocurrencia,
- $fd04 : fd4_S =$ media,
- $fd05 : fd5_S =$ desv. std,
- $fd06 : fd6_S =$ homogeneidad-coocurrencia,
- $fd07 : fd7_I =$ media,
- $fd08 : fd8_I =$ desv. std,
- $fd09 : fd9_I =$ homogeneidad-coocurrencia.

Es dentro de la evaluación de la función de aptitud, F, donde con base a los descriptores evolucionados en cada proceso evolutivo (1 y 2), que se toma el análisis CBIR sobre toda la base de imágenes que en resumen por cada punto aleatorio sobre la imagen se abre una ventana de 10×10 pixeles, quedando 3 ventanas por punto (una ventana por capa en espacio de color HSI), y sobre cada ventana considerada por separado se calcula el descriptor elegido, formando un

patrón con 3 entradas por descriptor por ventana, tal que si son 3 capas y 3 descriptores, el patrón queda de dimensión 1×9, se pasan todos los patrones así formados a organizarse por medio del algoritmo $K-$medias, y finalmente aplicando un clasificador de distancia $k-$NN, se toman a los k vecinos más próximos. Más detalle se muestra en [6]. Un aspecto importante de la consideración del cambio de espacio de color a HSI, es que se tiene información de textura de la imagen y no del color. A continuación se describe detalladamente cada una de estas fases.

Fase 1. Primero, se configuran los parámetros del algoritmo de programación genética. Después, se genera la población de soluciones inicial para el proceso evolutivo. Cada solución se forma por un par de elementos. El primer elemento representa la combinación matemática de uno o más de los descriptores mencionados antes. Este elemento es sintetizado por el algoritmo, utilizando el conjunto de funciones y terminales que se ilustran más arriba; mientras que el segundo elemento representa un solo descriptor seleccionado al azar de entre los 9 propuestos. Posteriormente se evalúa el desempeño de cada una de estas soluciones, mediante el proceso de reconocimiento de imágenes a partir de su propio contenido que se propuso en el trabajo publicado en [6]. Las soluciones mejor evaluadas son seleccionadas para llevar a cabo la generación de la nueva población mediante los métodos de cruzamiento y mutación. Es necesario mencionar que la mejor solución se pasa directamente a la nueva población. Este proceso se lleva a cabo por un determinado número de generaciones, hasta que la mejor solución de la población no cambie. Una vez que el proceso evolutivo se detiene, se guarda la mejor solución para un uso posterior en la fase 2. La fase 1 se lleva a cabo 3 veces, obteniendo así 3 soluciones ganadoras de cada ejecución de esta fase.

Fase 2. El proceso evolutivo es similar al de la fase 1. En esta caso la población de soluciones se conforma de un solo elemento. Este elemento es sintetizado por el algoritmo evolutivo de esta fase, utilizando el conjunto de funciones y terminales que se ilustran en la Tabla 1. El proceso de evaluación en esta fase consiste nuevamente en aplicar el proceso de reconocimiento de imágenes a partir de su propio contenido que se propuso en el trabajo de [6]. La diferencia es, que se fabrican 3 tripletas cada vez. Una por cada una de las soluciones ganadoras de la fase previa; añadiendo a cada una el elemento sintetizado en esta fase. Por lo tanto, se obtienen tres evaluaciones por cada individuo; así que se conserva la mejor evaluación y se registra la tripleta que obtuvo dicha evaluación para que, al termino del proceso evolutivo se conozca la tripleta fabricada con el individuo ganador. Esta tripleta representa la solución al problema.

4. Experimentos y resultados

Los experimentos consistieron en probar el algoritmo que se detalló en la sección anterior sobre la imágenes de la base de datos de paisajes naturales desarrollada en [11], con resolución de 720×480 pixeles. De esta base de datos

Fig. 2. Ejemplos de las imágenes de los 5 escenarios naturales considerados, las pruebas se hicieron sobre dos tipos o clases

se tomaron 2 clases de imágenes, con 20 imágenes por clase como se muestra en la Figura 2. La metodología descrita en el capítulo anterior se implementó utilizando MatLab©, con el módulo de PG conocido como GPLab, el cual fue desarrollado en [7]. La Tabla 1 muestra el valor de los parámetros utilizados para el algoritmo de GP durante las pruebas experimentales. Los valores adecuados para cada parámetros se determinaron mediante prueba y error.

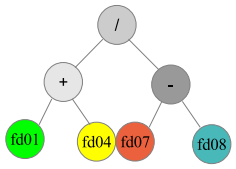
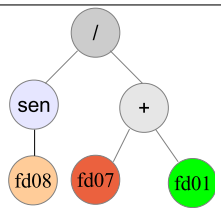
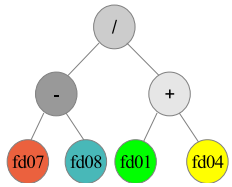
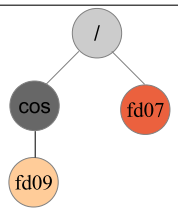
Tabla 1. Parámetros utilizados para el algoritmo basado en GP. Dichos parámetros están ordenados de acuerdo al módulo de GPLab al que pertenecen

Módulo Población inicial	
Población inicial	Mitad árboles completos y Mitad árboles de distribución aleatoria
Módulo Generación	
Generaciones	300
Tamaño de la población	100 individuos
Probabilidad de cruzamiento	0.7
Probabilidad de mutación	0.3
Tipo de Selección	Torneo de presión lexicográfica
Tipo de elitismo	Conservación del mejor por generación
Módulo Manejo de los individuos	
Profundidad de los árboles	Dinámica
Profundidad máxima dinámica	3 niveles
Profundidad máxima real	4 niveles

4.1. Análisis de resultados y comparación con otros resultados

Se realizaron 3 corridas. Los resultados que se hallaron de clasificación para cada una de estas corridas se muestran en la Tabla 2. Se puede apreciar que con los ajustes evolutivos para la PG se ha logrado un nivel de clasificación cercano al 100 % sobre 2 clases de la DB, 97.13 % y 96.63 %, con una complejidad de operadores bastante sencilla como los mostrados en las dos tripletas $\{descriptor_1, descriptor_2, descriptor_3\}$, con apenas un espacio de exploración de funciones de 6 tipos, con una profundidad y amplitud de árboles controladas.

Tabla 2. Soluciones encontradas en cada una de las tres corridas que se llevaron a cabo. Se presentan los árboles sintácticos de cada uno de los descriptores que conforman de la tripleta, así como el porcentaje de clasificación que alcanzan

Corrida	Descriptor 1	Descriptor 2	Descriptor 3	Desempeño obtenido
1				**97.13** %
2				**96.63** %

Comparando con los resultados obtenidos en [5,6], estamos muy próximos a igualarlos por menos del 3 %, y por encima de lo reportado por [11], pero considerando la mejora del clasificador de distancia por análisis local de textura. En cada uno de los 3 trabajos mencionados anteriormente, los descriptores fueron construidos manualmente acorde a la experiencia del procesamiento digital de imágenes, pero en nuestra metodología evolutiva propuesta estamos hallando nuevas soluciones en descriptores que no se habían considerado anteriormente en la literatura, con la limitante de la programación estructurada.

5. Conclusiones y trabajo futuro

En este artículo hemos presentado una primera aproximación, hasta ahora no documentada, de aplicar la evolución de descriptores estadísticos por medio

de la PG hacia la metodología CBIR, aplicada al problema de clasificación de escenarios naturales. Hasta este momento la exploración ha dado un resultado que podemos considerarlo muy bueno tras resolver el reto de la programación del paradigma de la coevolución desde una plataforma que per sé no la permite sobre un algoritmo evolutivo, dado el fenómeno de *bloat* documentado en la literatura, alcanzado unos descriptores robustos para dos escenarios naturales de 5 clases, que si vemos en la literatura, el problema no se resuelve satisfactoriamente por segmentación de las imágenes, pero sí por el análisis del contenido de las imágenes por medio de CBIR.

En un trabajo futuro se considera la coevolución con descriptores estadísticos que consideren la transformación hacia otros espacios de color, o bien de ser posible tomar otras bandas, e.g., el infrarrojo, para un problema en específico, con la idea de poder explorar evolutivamente soluciones más robustas para usar una técnica probada de clasificación de imágenes, pero sin importar el tipo de imágenes. Adicionalmente, será necesario trasladar la programación hacia un lenguaje de programación de bajo nivel, como ANSI C/C++, a fin de lograr mayor rapidez y precisión; y todo esto para lograr probar sobre más clases de imágenes.

Referencias

1. Benavides-Alvarez, C., Villegas-Cortez, J., Roman-Alonso, G., Aviles-Cruz, C.: Reconocimiento de rostros a partir de la propia imagen usando tecnica cbir. In: X Congreso Espanol sobre Metaheuristicas, Algoritmos Evolutivos y Bioinspirados MAEB 2015. pp. 733–740. Universidad de Extremadura, Merida, Extremadura. Spain (Jan 2015)
2. Deb, K.: Multi-Objective Optimization using Evolutionary Algorithms. Wiley Publishing (2001)
3. Fukunaga, K.: Introduction to statistical pattern recognition (2nd ed.). Academic Press Professional, Inc., San Diego, CA, USA (1990)
4. Koza, J.: Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press (1992)
5. Perez-Pimentel, Y., Osuna-Galan, I., Villegas-Cortez, J., Aviles-Cruz, C.: A genetic algorithm applied to content-based image retrieval for natural scenes classification. In: 13th Mexican International Conference on Artificial Intelligence (MICAI). pp. 155–161 (Nov 2014)
6. Serrano-Talamantes, J.F., Aviles-Cruz, C., Villegas-Cortez, J., Sossa-Azuela, J.H.: Self organizing natural scene image retrieval. Expert Systems with Applications 40(7), 2398–2409 (2013), http://www.sciencedirect.com/science/article/pii/S0957417412011888?v=s5
7. Silva, S., Almeida, J.: Gplab-a genetic programming toolbox for matlab. In: In Proc. of the Nordic MATLAB Conference (NMC-2003. pp. 273–278 (2005)
8. Smeulders, A.W.M., Worring, M., Santini, S., Gupta, A., Jain, R.: Content-based image retrieval at the end of the early years. IEEE Transactions on Pattern Analysis and Machine Intelligence 22(12), 1349–1380 (Dec 2000)
9. USNSF: US NSF workshop visual information management systems workshop report. In: Conference on Storage and Retrieval for image and Video Databases. Springer Secaucus, Denver, Colorado, USA (1992)

10. Villegas-Cortez, J., Olague, G., Aviles, C., Sossa, H., Ferreyra, A.: Automatic synthesis of associative memories through genetic programming: A first co-evolutionary approach. In: Applications of Evolutionary Computation, Lecture Notes in Computer Science, vol. 6024, pp. 344–351. Springer Berlin / Heidelberg (2010)
11. Vogel, J., Schiele, B.: Semantic modeling of natural scenes for content-based image retrieval. International Journal of Computer Vision 72(2), 133–157 (2006), http://dx.doi.org/10.1007/s11263-006-8614-1

Análisis del algoritmo de optimización por enjambre de partículas por medio de una aplicación gráfica 3D

Charles F. Velázquez Dodge, M. Mejía Lavalle

Centro Nacional de Investigación y Desarrollo Tecnológico,
Cuernavaca, Morelos,
México

{charlesdodge, mlavalle}@cenidet.edu.mx

Resumen. El método de optimización por enjambre de partículas, se ha vuelto popular en los últimos años, dado a su eficiencia y bajo costo computacional. En este documento, se realiza un breve análisis del algoritmo estándar por medio de una aplicación gráfica 3D que se desarrolló. El código fuente de la aplicación es libre y es posible descargarla en GitHub.

Palabras clave: Métodos de optimización, optimización de enjambre de partículas, algoritmos.

Analysis of Particle Swarm Optimization Algorithm Using a 3D Application

Abstract. Particle swarm optimization method has become popular in recent years, because of its efficiency and low computational cost. In this document, a brief analysis of the standard algorithm is performed by a 3D application. The source code of the application is free software and you can download at GitHub.

Keywords: Optimization methods, particle swarm optimization, algorithms.

1. Introducción

El algoritmo de optimización por enjambre de partículas (PSO), fue originalmente desarrollado por James Kennedy y Russell Eberhart en 1995, es un algoritmo del área de la inteligencia artificial de la rama de inteligencia de enjambres. Está inspirada en el comportamiento social de los seres vivos [1]. Comparado con los algoritmos genéticos (GA) que se basa en el mecanismo de evolución biológica, el algoritmo de optimización por enjambre de partículas se inspira en la evolución en el comportamiento colectivo, principalmente trata de imitar el comportamiento social de varios grupos de animales como lo son los cardúmenes, parvadas, manadas, etc. [2]. Los dos algoritmos se basan en población de individuos, solo que con diferentes enfoques y han demostrado ser eficientes para la solución de problemas complejos.

Ambos son algoritmos de optimización metaheurísticos, ya que utilizan analogías con otros procesos para resolver el problema, los métodos metaheurísticos no se especializan a resolver un problema en particular, por lo que puede usarse para cualquier problema. Estos métodos no garantizan dar el mejor resultado, pero sí un resultado aceptable [3].

Otra característica de estos algoritmos, es que son no deterministas (estocásticos), esto quiere decir que los resultados obtenidos no siempre serán los mismos aunque se trate de una misma función.

En muchas de sus aplicaciones, tanto los algoritmos genéticos, como los algoritmos de enjambre de partículas, ofrecen resultados de calidad del 99%, sin embargo, se ha demostrado que los algoritmos de enjambre de partículas son superiores en cuanto a eficiencia, debido a su bajo costo computacional[4].

El algoritmo de optimización de enjambre de partículas original ha tenido varios cambios y han surgido variaciones del mismo según el problema que se quiera resolver. Sin embargo, en 2006, se trató de establecer un estándar (SPSO), y que posteriormente se le ha contribuido con algunas sugerencias y cambio en el 2007 y 2011 [5-6]. Comparado con el algoritmo clásico, el estándar, agregó el factor social, cognitivo, de inercia y constricción. También se cambió el modo en que las partículas se comunicaban por medio de topologías definidas.

2. Descripción del estándar

En esta parte se describirán los factores utilizados en el estándar, así también como las fórmulas utilizadas según los factores a utilizar.

2.1. Cognitiva y social

La cognitiva contribuye para que la partícula tenga una especie de memoria y pueda saber si anteriormente había adoptado una mejor posición [3].

El factor social, es el responsable de que la partícula sea influenciado por otras partículas en una mejor posición, provocando ser atraída al óptimo encontrado [3].

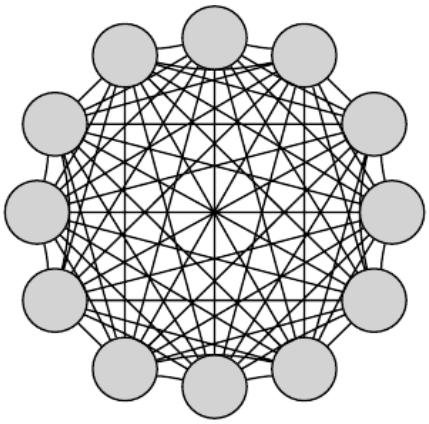


Fig. 1. Modelo gbest [6]

2.2. Topología gbest y lbest

El concepto del modelo gbest es que todas las partículas se comunican entre sí y el mejor punto encontrado, es comunicado a las demás partículas. Tiene la ventaja de ser muy rápido para encontrar un valor óptimo, pero esto genera un problema, ya que puede provocar una convergencia prematura en un óptimo local, que queremos evitar en la mayoría de los casos [7].

En contraste, el modelo lbest es más lento pero hay más posibilidad de evitar una convergencia prematura, en la práctica, el modelo mantiene varios puntos de atracción que hace escapar de óptimos locales [7].

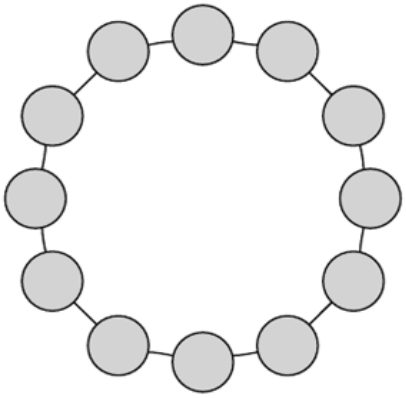


Fig. 2. Modelo lbest [6]

2.3. Inercia y constricción

La inercia es un factor que se añadió para dar mayor estabilidad a la fórmula, permite que la partícula siga una trayectoria constante ignorando la influencia de otros [3].

El factor de constricción, a diferencia de la inercia, es variable, permite un equilibrio entre búsquedas locales y globales. Cuando este factor es menor a 4, el enjambre se mueve lentamente y tiene una convergencia lenta, mientras que si es mayor a 4, logra una convergencia rápida [7].

El factor de inercia modulada, brinda un comportamiento similar al de la constricción. Consiste en comenzar con una inercia alta, e ir disminuyendo con el paso del tiempo [7]. Está comprobado que este factor da mejores resultados que una inercia constante [3].

2.4. Fórmulas

A continuación plantearemos las ecuaciones que conforman el algoritmo. Principalmente existen dos ecuaciones según el factor que se ha elegido, ya sea el factor de inercia o el de constricción. La notación utilizada es la siguiente:

v= velocidad,

i= índice de la partícula,
k= estado en el tiempo,
C_1 = cognitiva,
rand = valor aleatorio,
p^i = mejor posición de la partícula,
C_2 = social,
p^g = mejor posición global,
x = posición de la partícula,
ω= inercia.

La fórmula para el cálculo de velocidad de cada partícula utilizando inercia es:

$$v_{k+1}^i = \omega v_k^i + c_1 rand\left(p_k^i - x_k^i\right) + c_2 rand\left(p_k^g - x_k^i\right). \quad (1)$$

Por otro lado para usar el factor de constricción, la fórmula es:

$$v_{k+1}^i = \chi\left[v_k^i + c_1 rand\left(p_k^i - x_k^i\right) + c_2 rand\left(p_k^g - x_k^i\right)\right], \quad (2)$$

donde χ es el valor de constricción determinado por:

$$\chi = \frac{2}{\left|2 - \zeta - \sqrt{\zeta^2 - 4\zeta}\right|}, \quad (3)$$

$$\zeta = c_1 + c_2. \quad (4)$$

Por último la fórmula para actualizar su posición utilizando inercia es:

$$x_{k+1}^i = x_k^i + v_k^1. \quad (5)$$

Algunos autores proponen otras mejoras como es el de dividir entre un incremento del tiempo, [4] utilizar valores aleatorios basados en alguna distribución [5], sin embargo, en estos momentos nos basaremos en el estándar del 2007.

3. Contribución

Se contribuye con una aplicación que ayudará al entendimiento del algoritmo, fue desarrollada en el lenguaje de programación C++ con librerías multiplataforma Qt y QwtPlot3D. La aplicación se ha probado en varios equipos y funciona perfectamente en un sistema GNU/Linux, sin embargo se han encontrado bugs al momento de ejecutar la aplicación de Windows. Todo el software utilizado es software libre y la aplicación desarrollada se ha subido a GitHub para que sea accesible para todo el público. El código se encuentra en:

https://github.com/CharlesVD/Optimizacion_por_Enjambre_de_Particulas

Las librerías de Qt permiten el desarrollo de aplicaciones gráficas en 3D gracias a que tiene incorporado OpenGL, sin embargo se optó por usar QwtPlot3D para agilizar el desarrollo y no generar primitivas 3D desde cero.

4. Funciones de prueba en la aplicación

Para la aplicación tomamos varias funciones de la tesis doctoral de Helbert Eduardo Espitia Cuchango [7].

Tabla 1. Funciones

Nombre	Función	Mínimo
Parabolic	$f(x,y) = x^2 + y^2$	0
Ackley	$f(x,y) = e + 20 - 20 \exp(-0.2\sqrt{0.5x^2 + y^2}) - \exp(0.5(\cos(2\pi x) + \cos(2\pi y)))$	0
Circles	$f(x,y) = (x^2 + y^2)^{0.25} ((\sin(50(x^2 + y^2)^{0.1}))^2 + 1)$	0
Rastrigin	$f(x,y) = 20 + (x^2 - 10\cos(2\pi x) + y^2 - 10\cos(2\pi y))$	0
Equal Peaks	$f(x,y) = \cos(x)^2 + \sin(y)^2$	0
Himmelblaus	$f(x,y) = -0.01 (200 - (x^2 + y^2 - 11)^2 - (x + y^2 - 7)^2)$	-2
Peaks	$f(x,y) = 3(1-x)^2 e^{-(x^2+(y+1)^2)} - 10(\frac{x}{5} - x^3 - y^5)e^{-(x^2+y^2)} - \frac{1}{3}e^{-((x+q)^2+y^2)}$	-6.4169
Schaffer	$f(x,y) = 0.5 + \frac{\sin(\sqrt{x^2 + y^2})^2 - 0.5}{(1 + 0.1(x^2 + y^2))^2}$	0
Rosenbrock (banana)	$f(x,y) = 100(y - x^2)^2 + (1 - x)^2$	0

5. Conclusión

Gracias a la aplicación se logró entender y analizar el algoritmo, se desarrolló una aplicación que podrá ser útil en generaciones futuras que quieran experimentar o analizar de una manera visual e intuitiva, el comportamiento del algoritmo de optimización por enjambre de partículas. Al ser una aplicación libre, es posible analizarla y modificarla para incorporar otras variaciones del algoritmo, la interfaz gráfica y el algoritmo están separados en clases por lo que no es muy difícil hacer los cambios.

6. Trabajo a futuro

Se pretende incorporar más variaciones del algoritmo a la aplicación, mejorar la compatibilidad con otros sistemas operativos, añadir más parámetros de configuración y durante el proceso, ir encontrado posibles mejoras al algoritmo.

Referencias

1. Kennedy, J., Eberhart, R.: Particle Swarm Optimization. Purdue School of Engineering and Technology Indianapolis (1995)
2. Kennedy, J., Eberhart, R.: Tutorial on Particle Swarm Optimization. In: IEEE Swarm Intelligence Symposium (2005)
3. Benítez, R., Escudero, G., Kanaan, S.: Inteligencia artificial avanzada. Universitat Oberta de Catalunya, pp. 164–199 (2013)
4. Hassan, R., Cohanim, B., de Weck, O., Venter, G.: A copmarison of Particle Swarm Optimization and The Genetic Algorithm. Massachusetts Institute of Technology, Vanderplaats Research and Development (2004)
5. Clerc, M.: Standard Particle Swarm Optimization (2012)
6. Bratton, D., Kennedy, J.: Defining a Standard for Particle Swarm Optimization. In: IEEE Swarm Intelligence Symposium (2007)
7. Espitia Cuchango, H.E.: Algoritmo de optimización basado en enjambres de partículas con comportamiento de vorticidad. Universidad Nacional de Colombia, Facultad de Ingeniería, pp. 11–22 y pp. 92–96 (2014)

Algoritmo de optimización mediante forrajeo de bacterias híbrido para el problema de selección de portafolios con restricción de cardinalidad

Christian Leonardo Camacho-Villalón[1], Abel García-Nájera[2], Miguel Ángel Gutiérrez-Andrade[1]

[1] UAM Iztapalapa, Departamento de Ingeniería Eléctrica, Ciudad de México, México

[2] UAM Cuajimalpa, Departamento de Matemáticas Aplicadas y Sistemas, Ciudad de México, México

clcamachov@gmail.com, gamma@xanum.uam.mx, agarcian@correo.cua.uam.mx

Resumen. Este trabajo aborda el problema de la selección de portafolios de inversión óptimos (PSP). Mucha investigación se ha hecho en torno a esta tema, la mayor parte de los trabajos han buscado extender el modelo de Markowitz considerando restricciones realistas (piso-techo, clases y cardinalidad), y/o introduciendo otras medidas de riesgo (semivarianza, desviación absoluta, valor en riesgo, etc.). En este documento presentamos los resultados preliminares de un algoritmo de optimización multiobjetivo híbrido basado en optimización por forrajeo de bacterias (BFO), al cual integramos el enfoque de aprendizaje incremental basado en probabilidad (PBIL). El enfoque de PBIL hace uso de información estadística para guiar el proceso de mejora incremental de las bacterias. Para mejorar el desempeño de BFO, implementamos una función lineal decreciente para el tamaño de los pasos quimiotácticos, reinicialización de las bacterias y asignación de pesos aleatorios durante la fase de reproducción. Nuestra formulación incluye las restricciones de cardinalidad y piso-techo, dos restricciones realistas que son necesarias en la mayoría de los mercados bursátiles del mundo. Basados en el modelo de media-varianza propuesto por Markowitz, utilizamos la bien conocida formulación de Frontera Eficiente (EF) que integra en un solo objetivo el riesgo y el retorno a través de un parámetro de aversión al riesgo. Con la formulación anterior y utilizando un conjuntos de datos estándar para el PSP, llevamos a cabo la evaluación del desempeño del algoritmo. Los resultados obtenidos mostraron que nuestro algoritmo es capaz encontrar soluciones de buena calidad distribuidas uniformemente sobre la frontera eficiente.

Palabras clave: Optimización de portafolios, selección de portafolios, optimización por forrajeo de bacterias, BFO, inteligencia de enajambre, aprendizaje incremental.

Christian Leonardo Camacho-Villalón, Abel García-Nájera, Miguel Ángel Gutiérrez-Andrade

Hybrid Bacterial Foraging Optimization Algorithm for the Cardinality Constrained Portfolio Selection Problem

Abstract. In this paper we tackle the optimal portfolio selection problem (PSP). Many research has been made around this subject mainly in two ways, whether extending the Markowitz model by taking into account real-world constraints (floor-ceiling, class and cardinality) or introducing different risk measures like semivariance, value at risk, absolute desviation, etc. Here, we present the preliminary results of a new multiobjective heuristic based in the bacterial foraging optimization (BFO) which integrates the population based incremental learning (PBIL) approach. PBIL uses statistical information to guide the optimization process in the bacteria population. Furthermore, to improve the BFO heuristic we introduced a lineal decreasing function for the chemotaxis steps size, bacterias reinitialization and random weighing in the reproduction step. Our formulation include the cardinality and floor-ceiling constraints, both are real-world constraints needed in most of the stock markets. Based in the mean-variance model (first proposed by Markowitz) we used the well-known efficient frontier formulation which introduces a risk aversion parameter to weigh between risk and mean return, leading into a single-objective formulation problem. Applying our algorithm to solved the above mentioned model, we performed tests with a standard dataset taken from the OR-Lib. The experimental results shown our algorithm is able to find good-quality solutions uniformly distributed over the real efficient frontier.

Keywords: Portfolio optimization, portfolio selection, bacterial foraging optimization, BFO, swarm intelligence, population based incremental learning

1. Introducción

En 1952 Harry Markowitz hizo la mayor contribución sobre el problema de la selección de portafolios (PSP) con la publicación del modelo de media-varianza [1], también conocido como el modelo de Markowitz. Este modelo involucra dos objetivos en conflicto, por un lado se busca maximizar la ganancia (media) y por otro minimizar el riesgo (varianza), resultando en un problema de programación cuadrática (QP) de gran escala [2]. El modelo de media-varianza de Markowitz es ampliamente utilizado, sin embargo, hace una serie de simplificaciones y suposiciones irreales [3], entre las que están: 1) un mercado perfecto en donde no hay impuestos, 2) no considera costos de transacción, 3) la venta en corto no está permitida y 4) los activos se pueden dividir de manera infinita para su comercialización. Al extender el modelo original para incluir restricciones prácticas que son relevantes (esto es, hacerlo más realista), se vuelve más complicado de resolver. Si se incluye en la formulación del problema alguna restricción

que implique números enteros (como la restricción de cardinalidad o la de lotes mínimos), el problema se transforma de uno de programación cuadrática (QP) a uno de programación entera mixta cuadrática (QMIP), que está probado es de tipo NP-difícil [4]. De igual manera, si hay por lo menos una restricción de tipo cuadrático, es necesario recurrir a técnicas de optimización alternativas.

El PSP se puede formular como un problema de optimización multiobjetivo, en este tipo de problemas ya no se busca obtener una única solución, sino un conjunto de soluciones que representen el mejor compromiso entre todos los objetivos del problema. Las técnicas de optimización multiobjetivo tienen la habilidad de manejar de manera simultánea un conjunto de soluciones llamada población. Al conjunto de soluciones eficientes de la población se les llama óptimos de Pareto. Una característica esencial que se busca en los problemas multiobjetivo es lograr una distribución uniforme de las soluciones eficientes sobre el frente de Pareto.

Existen diversas técnicas matemáticas y métodos analíticos para resolver el problema de la selección de portafolios [5], sin embargo, la eficacia de estos métodos es limitada al no considerar restricciones realistas a la formulación del problema. El análisis utilizando en estas técnicas generalmente tiene que "adaptar" el problema para que pueda ser resuelto. Al considerar un número grande de activos en el problema, las técnicas analíticas se pueden ver rebasadas, además de volverse muy complicadas de emplear con un número grande de restricciones en el modelo o ser de tipo cuadrático. Por otro lado, las técnicas metaheurísticas pueden hacer frente a estos inconvenientes y encontrar la frontera eficiente con restricciones [6]. Dentro de las técnicas metaheurísticas están el algoritmo de recocido simulado (SA) y búsqueda tabú (TS). También se han empleado técnicas híbridas basadas en búsqueda local (LS) y el procedimiento de programación cuadrática (QP), los cuales han mostrado resultados comparables o superiores a los soluciones matemáticas y los métodos analíticos.

Muchos trabajos han utilizado algoritmos basado en poblaciones estocásticas, dentro de éstos, los algoritmos genéticos (GA) han mostrado mejores resultados que SA y TS [7]. Una técnica híbrida que utiliza un LS para encontrar el número óptimo de activos y después QP para determinar el peso de cada uno en el portafolio mostró buenos resultados [9]. La optimización multiobjetivo por colonia de hormigas (ACO) [8] se ha presentado como una metaheurística especialmente efectiva, los resultados obtenidos con esta técnica son comparables a los que se obtienen con la optimización de Pareto por recocido simulado y el algoritmo NSGA. El uso de un modelo híbrido de una red neuronal artificial con el algoritmo de optimización por enjambre de partículas (PSO) mostró la flexibilidad de las técnicas híbridas, así como su superioridad en predecir el desempeño del portafolio [6].

La optimización por forrajeo de bacterias (BFO) fue propuesta originalmente por Passino [10] en 2002 y es parte de las técnicas de inteligencia de enjambre (SI). Las bacterias en el algoritmo de BFO implementan un tipo de caminata aleatoria influenciada para encontrar las mejores soluciones. El algoritmo sigue la estrategia de forrajeo (alimentación) de bacterias reales en tres aspectos:

dirigirse hacía las regiones donde están las mejores soluciones y permanecer ahí más tiempo, evadir las regiones con las peores soluciones y salir de las regiones donde no se puedan mejorar las soluciones. Utilizando este comportamiento, el algoritmo propuesto tiene la habilidad de exploración y explotación del espacio de búsqueda para encontrar la frontera eficiente. Por otro lado, PBIL utiliza la idea evolutiva de una población de individuos basada en información estadística recolectada durante el proceso evolutivo.

El algoritmo propuesto integra a BFO la técnica de PBIL, así como algunas mejoras al algoritmo de BFO que ayudan a una adecuada exploración y explotación del espacio de búsqueda.

El resto del documento está estructurado de la siguiente manera. En la Sección 2 se describe el modelo de media-varianza y la formulación de Frontera Eficiente. En la Sección 3 se describen los algoritmos de BFO y PBIL. En la Sección 4 se introduce el algoritmo propuesto y las mejoras. En la Sección 5 se discuten los resultados obtenidos por el algoritmo. Finalmente, en la Sección 6 aparecen las conclusiones y el trabajo futuro.

2. El problema de selección de portafolios

2.1. Formulación de Frontera Eficiente

El modelo clásico de media-varianza de Markowitz [1] busca de manera simultánea la minimización del riesgo y la maximización del retorno esperado considerando como restricción que la suma de todos los activos debe ser igual a uno. Una de las formulaciones más utilizadas que emplean la formulación clásica de media-varianza es la siguiente:

$$\begin{aligned} \text{minimizar} \quad & \lambda\left[\sum_{i=1}^{N}\sum_{j=1}^{N} x_i x_j \sigma_{ij}\right] - (1-\lambda)\left[\sum_{i=1}^{N} x_i r_i\right] , \\ \text{sujeto a} \quad & \sum_{i=1}^{N} x_i = 1 , \\ & 0 \leq x_i \leq 1, \quad i = 1, \ldots, N . \end{aligned} \tag{1}$$

El modelo integra en un solo objetivo el riesgo y el retorno. Es posible encontrar diferentes valores para la función objetivo variando el retorno esperado deseado $R^{\star} = \sum_{i=1}^{N} x_i r_i$. La forma más común de hacerlo es introduciendo un factor de aversión al riesgo $\lambda \in [0, 1]$. Con este nuevo parámetro λ el modelo puede ser descrito a través de una sola función objetivo.

Cuando λ es cero, el modelo maximiza el retorno esperado del portafolio sin considerar la varianza (riesgo). En cambio, cuando λ es igual a uno, el modelo minimiza el riesgo del portafolio sin tomar en cuenta el retorno esperado. La sensibilidad del inversionista al riesgo se incrementa al incrementarse λ. Para diferentes valores de λ se obtienen diferentes valores de la función objetivo. Si se traza la intersección entre el valor del retorno esperado y la varianza para los

diferentes valores de λ se obtiene una curva continua llamada frontera eficiente, en donde cada punto de la frontera eficiente indica un valor óptimo.

Las dos restricciones realistas que más frecuentemente se han utilizado para el problema de optimización de portafolios de inversión son las siguientes:

i) *Piso-techo*: Imponen los límites inferiores y/o superiores (ϵ , δ) para el peso de los activos en lugar de utilizar cero como mínimo y uno como máximo. Por lo tanto, un activo no puede representar menos o más de cierta proporción del total del capital a invertir.

ii) *Cardinalidad*: Obligan a que los activos seleccionados en el portafolio respeten ciertas restricciones. Existen dos versiones de esta restricción. La primera versión (exacta) impone que el número de bonos seleccionados sea igual a un valor K . La segunda versión (suave) imponen los límites inferior y superior (Z_L, Z_U) para este valor.

La formulación del problema de la selección de portafolios con restricción de cardinalidad (CCPS) y piso-techo es la siguiente:

$$\begin{aligned} \text{minimizar} \quad & \lambda\left[\sum_{i=1}^{N}\sum_{j=1}^{N} x_i x_j \sigma_{ij}\right] - (1-\lambda)\left[\sum_{i=1}^{N} x_i r_i\right] , \\ \text{sujeto a} \quad & \sum_{i=1}^{N} x_i = 1 , \\ & \sum_{i=1}^{N} z_i = K , \\ & \epsilon z_i \leq x_i \leq \delta z_i, \quad i = 1, \ldots, N , \\ & z_i \in [0,1], \quad i = 1, \ldots, N . \end{aligned} \tag{2}$$

donde la variable z_i es de tipo binario y permite saber si el activo i está presente en la solución. El problema que resuelve nuestro algoritmo de optimización es el que se encuentra formulado en (2).

3. Algoritmos híbrido BFO-PBIL

3.1. BFO

El algoritmo original de optimización por forraje de bacterias (BFO) fue propuesto por Kevin M. Passino [10] en 2002 y es uno de los métodos más recientes dentro del área de inteligencia de enjambre (SI) para la optimización de problemas continuos. El algoritmo imita el comportamiento de forrajeo que llevan a cabo las bacterias de Escherichia coli (E. coli) presentes en el intestino humano. Las bacterias artificiales realizan tres actividades de forrajeo básicas: quimiotaxis, reproducción y eliminación-dispersión. En un movimiento quimiotáctico, el enjambre de bacterias trata de moverse y permanecer en los

entornos ricos en nutrientes, abandonar las regiones pobres en nutrientes rápidamente y permanecer alejadas de los lugares peligrosos.

Una bacteria lleva a cabo un movimiento quimiotáctico en dos pasos: *nado* y *desplome*. Las bacterias pueden hacer varios nados en una misma dirección si la concentración de nutrientes se incrementa a su alrededor. Una vez que la bacteria detecta que los nutrientes a su alrededor disminuyen, ejecuta la acción de desplome para cambiar rápidamente la dirección de la búsqueda. Los pasos de nado y desplome se ejecutan de manera alternada, a través del nado las bacterias permanecen por mayor tiempo en las regiones ricas en nutrientes y mediante el desplome son capaces de salir rápidamente de las regiones poco atractivas. La quimiotaxis puede ser vista como una estrategia bacteriana de optimización local, cuyo comportamiento móvil se describe mediante la siguiente fórmula:

$$\Theta^i(j+1,k,l) = \Theta^i(j,k,l) + C(i) \times \Phi(j), \tag{3}$$

donde $\Theta^i(j,k,l)$ denota la posición de la bacteria i en el paso quimiotáctico j, el paso reproductivo k y el paso de eliminación-dispersión l. $C(i)$ es el tamaño del paso quimitáctico de la bacteria i, el vector $\Phi(j)$ se utiliza para definir la dirección del movimiento aleatorio de un movimiento de desplome en el paso quimiotáctico j.

Para producir nuevas soluciones, las bacterias realizan una serie de movimientos quimiotácticos en los cuales se incrementa y decrementan el peso de los activos presentes en el portafolio (a través de nado y desplome). Cada nueva solución es evaluada por el algoritmo, si la solución nueva es mejor que la solución actual, esta última es reemplazada en el enjambre de bacterias. La ecuación quimiotáctica está definida como sigue:

$$nxx_b = xx_b^t + C(j) \times \Delta D_b(j), \forall j, \tag{4}$$

donde nxx_b son los nuevos valores obtenidos para la bacteria b, t es el número de la iteración del paso quimiotáctico, $C(j)$ es una constante que representa el tamaño del movimiento quimiotáctico (controla la distancia del movimiento), y $\Delta D_b(j)$ es un número aleatorio en el intervalo $[-1,1]$ que denota que la magnitud del cambio en la dirección en un paso de desplome. Tanto el nado como el desplome utilizan constantes para indicar el tamaño del paso, $C_d(j)$ indica el valor de $C(j)$ para el desplome y $C_n(j)$ el valor para el nado. Un paso quimiotáctico incluye un desplome y número de nados, el algoritmo incluye un mecanismo de autorevisión que implementa cada bacteria para controlar la ocurrencia de estos pasos.

Después de ejecutar una serie de movimientos quimiotácticos las bacterias intentarán reproducirse para mejorar las probabilidades de supervivencia. Cada una de las bacterias más fuertes se reproduce dividiéndose asexualmente en dos bacterias, las bacteria recién creada se ubicarán cerca del padre. Al mismo tiempo, las bacterias más débiles mueren dejando el número de bacterias en la población constante (este proceso es similar a la selección en los GA).

Finalmente, debido a cambios repentinos o graduales en el entorno local, el evento de eliminación puede suceder de tal manera que un subconjunto del

enjambre de bacterias sea eliminado o forzado a moverse a otro lugar. Si una bacteria es eliminada, una nueva será generada y colocada de manera aleatoria en el espacio de búsqueda (esta operación es similar a la mutación en los GA). El proceso de dispersión se encarga de cambiar de lugar las bacterias existentes a una mejor región. Aunque la probabilidad de que ocurran los eventos de eliminación-dispersión es baja, después de un periodo largo de tiempo, este proceso incrementa la diversidad de las soluciones y mejora la búsqueda local (evitando quedar atrapado en mínimos locales).

3.2. PBIL

PBIL está basado en la idea evolutiva de una población de individuos basada en información estadística recolectada durante el proceso evolutivo. Asumiendo que no hay dependencia entre las variables (esto es, la selección de los activos son eventos mutuamente excluyentes), PBIL utiliza un vector de probabilidad para representar la distribución de todos los individuos. El vector de probabilidad adquiere aprendizaje hacia el vector que representa la mejor solución y se utiliza para generar la siguiente generación de individuos.

3.3. Mejoras al algoritmo BFO

Existen estudios recientes que buscan mejorar algunas características del algoritmo de BFO, con respecto a nuestra técnica de optimización vale la pena mencionar los siguientes trabajos. En [11] los autores agregaron un mecanismo de comunicación que emplea la fórmula de actualización de movimiento de PSO (*Gbest*), de esta manera las bacterias son guiadas hacia la mejor solución en cada iteración. Esta mejora está basada en el hecho de que otras técnicas de optimización (como la evolución diferencial (DE) y PSO) que hacen uso de la comunicación para aprender de las demás partículas a su alrededor, han mostrado buenos resultados y un mejora significativa en el desempeño del algoritmo.

Otra propuesta importante aparece en [12], donde los autores utilizaron una función decreciente linealmente para definir el tamaño de los pasos quimiotácticos hasta un valor fijo. De esta manera, las bacterias hacen cambios grandes al inicio del proceso de optimización y progresivamente se vuelven más pequeños. Esta mejora también tiene su justificación en el algoritmo de PSO y los coeficientes de aceleración utilizados para actualizar la posición de una partícula. Al igual que la técnica propuesta en [12], los valores disminuyen gradualmente de forma lineal. La fórmula propuesta por los autores para calcular el tamaño de los pasos quimitácticos es básicamente la misma que la de PSO.

Las dos técnicas anteriores ofrecen al algoritmo BFO original una mejoría notable en los resultados reportados por los autores. Respecto a la primera ([11]), en nuestro algoritmo BFO-PBIL mejorado la técnica de PBIl permite hacer uso de la información de las mejores soluciones de una manera muy eficiente sin necesidad de introducir cálculos adicionales para cada una de las bacterias. Es decir, después de un proceso quimiotáctico se identifica a la mejor y peor solución en la población, con esta información se actualiza el vector de probabilidad y se

completa la población de bacterias para el siguiente ciclo de optimización por quimiotaxis.

Respecto a la segunda técnica ([12]), en el algoritmo original de BFO el tamaño de los movimiento quimiotácticos de nado y desplome es constante durante toda la ejecución del algoritmo, sin embargo, se ha visto que si el tamaño es demasiado grande las bacterias pueden fallar en encontrar al óptimo global realizando numerosos nados. Por otro lado, si el tamaño del movimiento es muy pequeño es posible que a las bacterias les tome mucho tiempo encontrar el óptimo global. En nuestro algoritmo BFO-PBIL mejorado utilizamos este enfoque e implementamos una cantidad decreciente para el tamaño de la constante de desplome ($C_d(j)$) según [12].

3.4. Algoritmo BFO-PBIL mejorado

El algoritmo de optimización híbrido propuesto BFO-PBIL mejorado está inspirado principalmente en tres trabajos importantes y recientes [13], [16] [14].

Definiciones para el algoritmo BFO-PBIL mejorado:

λ = Factor de aversión al riesgo,
$C_{d_{max}}$ = Valor máximo para el tamaño del desplome,
$C_{d_{min}}$ = Valor mínimo para el tamaño del desplome,
N_B = Número de bacterias en la población,
N = Número de activos disponible,
v = Vector de probabilidad (PBIL),
ED_{max} = Número de movimientos de eliminación dispersión,
R_{max} = Número de movimientos reproductivos,
Q_{max} = Número de movimientos quimiotácticos,
G_{best} = La bacteria con mejor valor de aptitud,
G_{wort} = La bacteria con el peor valor de aptitud,
$Prob_{ED}$ = Probabilidad de eliminación-dispersión de un activo,
$Capital$ = Capital disponible para invertir,
ϵ = Límite inferior (restricción de piso-techo),
δ = Límite superior (restricción piso-techo),
K = Número de activos en el portafolio (restricción cardinalidad),
LR = Porcentaje de aprendizaje positivo,
NEG_LR = Porcentaje de aprendiza negativo.

Utilizamos el enfoque propuesto inicialmente por [7] dividiendo λ en 50 partes iguales. El valor del factor de aversión al riesgo λ en la iteración del algoritmo j se calcula con:

$$\lambda_j = (j-1)/49 \quad j = 1, ..., 50. \tag{5}$$

Tamaño de desplome decreciente: La función decreciente linealmente para el tamaño de la constante de desplome se determina con base en un valor inicial máximo ($C_{d_{max}}$) y un valor final mínimo ($C_{d_{min}}$), si Q_{max} es el número máximo

Algoritmo 1 BFO-PBIL mejorado

```
Inicializa vector de probabilidad incremental en 0.5
Inicializa población aleatoria inicial
para N_ed ← 1 to ED_max hacer
  para N_rep ← 1 to R_max hacer
    para N_quim ← 1 to Q_max hacer
      para b ← N_B/2) to N_B hacer
        Realiza movimientos de desplome y nado {4}
      fin para
    fin para
    Elimina la mitad de la población según f(b);
    Actualiza el vector v con (G_best) y (G_wort) {Ecs:78}
    Genera nueva bacteria b {Según el algoritmo:2}
  fin para
  para b ← 1 to N_B hacer
    si rand[0,1] ≤ Prob_ED entonces
      Elimina el Activo de la bacteria
      Selecciona un activo no incluido previamente
    fin si
  fin para
fin para
```

de pasos quimiotácticos y Q_{act} el número de la iteración actual, para el paso quimiotáctico j el tamaño de la constante de desplome $C_d(j)$ está dado por:

$$C_d(j) = C_{d_{min}} + \frac{Q_{max} - Q_{act}}{Q_{max}} \times \left(C_{d_{max}} - C_{d_{min}}\right). \tag{6}$$

Vector PBIL: El algoritmo que proponemos utiliza el enfoque de [16] para actualizar el vector de probabilidad (v). La actualización se realiza de acuerdo a un porcentaje de aprendizaje que puede ser positivo (LR) o negativo (NEG_LR). El porcentaje utilizado no solo controla la velocidad a la que el vector cambia para parecerse a la mejor solución, sino también la cantidad del espacio de búsqueda que será explorado. El uso de aprendizaje positivo y negativo tiene como objetivo aumentar la probabilidad de incluir los activo que contribuye a generar una buena solución y alejarse de los que no lo hacen.

$$v_i = v_i \times (1 - LR) + s_i^{G_{best}} \times LR, \tag{7}$$

donde $s_i^{G_{best}}$ es una variable binaria que permite saber si el activo i está presente en la mejor solución G_{best}. Si además sucede que el activo i está presente en $s_i^{G_{best}}$ y no lo está en la peor solución ($s_i^{G_{worts}}$) entonces:

$$v_i = v_i \times (1 - NEG\_LR) + s_i^{G_{best}} \times NEG\_LR. \tag{8}$$

En [16] los autores utilizaron el enfoque de mutación parcialmente guiada (PGM), en el cual en cada iteración del proceso evolutivo, cada dimensión del

vector de probabilidad se muta con una cierta probabilidad MP. Si el activo i es seleccionado se da igual oportunidad de mutarlo según un porcentaje de mutación (MR) o con el valor de la mejor solución $s_i^{G_{best}}$. En nuestro algoritmo decidimos utilizar el vector PBIL únicamente para guiar la selección de los activos que van a integrar las nuevas soluciones durante el proceso reproductivo como aparece en el Algoritmo 2.

Algoritmo 2 Reproducción con vector de probabilidad

```
1: para i ← 1 to N hacer
2:    si  rand[0, 1] < 0.5 y v_i > 0.5 entonces
3:       b_i = rand[0, 1] ∗ Capital
4:    sino
5:       si  bp_i > 0 entonces
6:          b_i = rand[0, 1] ∗ Capital
7:       fin si
8:    fin si
9:    Repara la bacteria b {Sección:3.5}
10: fin para
```

Reproducción con pesos aleatorios: Después de un proceso quimiotáctico viene un proceso de reproducción. En el algoritmo original de BFO cada bacteria se dividen asexualmente haciendo una copia idéntica de si misma, nosotros utilizamos un esquema de reproducción con el vector de probabilidad (v) para generar la mitad de la población faltante. El mecanismo de reproducción da igual oportunidad de seleccionar un activo presente en la bacteria padre o en el vector (v), el activo debe tener un peso mayor en v a 0.5 ó un peso mayor a 0 en la bacteria padre, si no se cumple alguno de estos criterios el activo se selecciona aleatoriamente cuando la bacteria es reparada. El peso asignado a los activos en las nuevas bacterias se distribuye aleatoriamente. Las nuevas bacterias estarán integradas por los activos de mayor calidad quedando ubicadas en regiones prometedoras del espacio de búsqueda.

Reinicialización aleatoria: Otra mejora incluida en nuestro algoritmo es la reinicialización de las bacterias después de un proceso quimiotáctico. Cada bacteria se evalúa para saber si logró modificar su valor de aptitud de manera significativa (con una diferencia de 10^{-5}). La idea es identificar a las bacterias que pueden estar atrapadas en un óptimo local. Con el objetivo de obtener una adecuada relación entre la *exploración* y *explotación*, si al llegar al número máximo de movimientos quimiotácticos durante un proceso de quimotaxis la bacteria no cambió de posición se reinicializa a una posición nueva aleatoria.

Algoritmo 3 Restriccion de cardinalidad

para $b \leftarrow 1$ **to** N_B **hacer**
  Ordena b según $f(b)$
  si $|b| > K$ **entonces**
    repetir
      Elimina el activo de menor peso
    hasta $|b| = K$
  fin si
  si $|b| < K$ **entonces**
    repetir
      Agrega un activo aleatoriamente
    hasta $|b| = K$
  fin si
fin para

3.5. Manejo de restricciones

Para cumplir con las restricciones de presupuesto, cardinalidad y piso-techo implementamos una proceso de reparación que evalúa y corrige cada bacteria. Primero se revisa que la cardinalidad de la solución sea igual a K según se expresa en el Algoritmo (3).

Posteriormente, una función disminuye hasta δ el peso de los activo que exceden el límite superior y aumenta hasta ϵ los que se encuentran por debajo de este valor.

$$x_i = \begin{cases} \delta, & \text{si } x_i > \delta \\ \epsilon, & \text{si } x_i < \epsilon. \end{cases} \tag{9}$$

Finalmente, una función de normalización de pesos es utilizada para cumplir con la restricción de capital. Esta función hace uso de un acumulador de capital excedente o sobrante en caso de que no sea posible decrementar o incrementar el peso de un activo sin violar la restricción de piso-techo. Después de la normalización se asigna el capital sobrante o faltante a los activos que pueden absorberlo. En la ecuación (10) el parámetro *Capital* representa el capital disponible por el inversionista, nx_i es el nuevo peso asignado al activo.

$$nx_i = \begin{cases} Capital \times \left(\frac{x_i}{\sum_i^N x_i} \right), & \text{si } \epsilon \leq nx_i \leq \delta \\ x_i, & \text{si } nx_i < \epsilon \text{ ó } nx_i > \delta. \end{cases} \tag{10}$$

4. Experimentación y resultados

4.1. Conjunto de datos

Para probar el desempeño del algoritmo utilizamos un conjunto de datos estándar propuesto inicialmente en [7]. Este conjunto de datos ha sido ampliamente utilizado y es reconocido como un marco de comparación para la

evaluación de algoritmos de optimización. Los archivos están disponibles en [15] y cada uno está conformado por el número de activos, el retorno estimado y la varianza de cada activo, y el coeficiente de correlación para cada pareja de activos i,j. Los activos incluidos en los archivos corresponde a los precios de cierre de cinco índices bursátiles: Hang Seng en Hong Kong (31 activos), DAX 100 en Alemania (85 activos), FTSE 100 en Reino Unido (89 acciones), S&P 100 en EE.UU. (98 activos) y Nikkei 225 en Japón (225 activos). Finalmente, para cada archivo de datos los autores proveen los puntos que conforman la frontera eficiente real.

4.2. Configuración del algoritmo

Los parámetros de configuración del algoritmo se establecieron en $Max_{\lambda} = 50$, los valores máximos y mínimos para los pasos quimiotácticos $C_{d_{max}} = 0.01$ y $C_{d_{min}} = 0.005$, el tamaño de población $N_B = 30$, el número de pasos de eliminación dispersión $MAX_{Elim-Disp} = 2$, reproductivos $MAX_{Reprod} = 20$, y una probabilidad de eliminación dispersión $Prob_{ED} = 0.25$. El número de pasos quimitácticos se fijo en $MAX_{Quim} = 30$, con un $Max_{nados} = 2$ después de un desplome. El *Capital* se fijó en $500,000$ con un límite inferior $\epsilon = 0.01$ y superior $\delta = 1$ para la restricción de piso-techo, para la de cardinalidad el valor de $K = 10$ según el enfoque de [7]. La velocidad de aprendizaje positivo fue $LR = 0.1$ y del negativo $NEG\_LR = 0.075$.

4.3. Resultados

Utilizamos el método de evaluación propuesto por [7] que mide la porcentaje de desviación horizontal y verticalmente de cada punto encontrado no dominado con la frontera eficiente real. Los resultados incluyen las siguientes medidas de desempeño: la media del porcentaje de desviación (MPD), la mediana del porcentaje de desviación (MedPD), el número de puntos no dominados y el tiempo total expresado en segundo. Se utilizó la misma configuración para cada conjunto de datos con los que se probó el algoritmo (Sección 4.2). Los resultados mostrados en la Tabla 1 son el promedio de veinte ejecuciones del algoritmo para los conjuntos de datos de 31, 85 y 89 activos resolviendo el PSP con restricciones de cardinalidad y piso-techo. En la Tabla 2 aparecen los mejores resultados obtenidos para el PSP sin restricciones.

En la Tabla 1 se presenta la comparación de BFO-PBIL contra PBILDE [16] que utiliza la técnica de evolución diferencial (DE) y tres heurísticas propuestas en [7] que incluyen un algoritmo genético (GA), búsqueda tabú (TS) y recocido simulado (SA). En la Figura 1 se muestra la frontera eficiente encontrada por nuestro algoritmo BFO-PBIL mejorado y la frontera eficiente real resuelta mediante programación cuadrática (QP). Los resultados que hemos obtenido hasta el momento para el PSP con restricciones son pobres comparados con las otras soluciones, creemos que esto es debido a una configuración deficiente en los parámetros del algoritmo. La razón por la que consideramos estos último, es que en las pruebas realizadas para el PSP sin restricciones el algoritmo mostró un

Tabla 1: Comparativa del desempeño para el PSP con restricciones

N	Medida	BFO-PBIL	PBIL-DE	Chang-GA	Chang-TS	Chang-SA
31	Puntos	276	**6367**	1317	1268	1003
	MPD(%)	4.3012789938	**0.6196**	0.9457	0.9908	0.9892
	MedPD(%)	4.4158656188	**0.4712**	1.1819	1.1992	1.2082
	Tiempo	759	113	172	**74**	79
85	Puntos	151	**3378**	1270	1467	1135
	MPD(%)	14.3790364757	**1.5433**	1.9515	2.5383	2.4675
	MedPD(%)	9.9511868431	**1.0986**	2.1262	3.0635	2.4299
	Tiempo	1406	1358	544	**199**	210
89	Puntos	203	**2957**	1482	1301	1183
	MPD(%)	7.9960532075	0.8234	0.8784	1.3908	**0.7137**
	MedPD(%)	7.2076477735	**0.5134**	0.5938	0.6361	1.1341
	Tiempo	1533	1496	573	246	**215**

Tabla 2: Comparativa del desempeño para el PSP sin restricciones

N	Medida	BFO-PBIL	PBIL-DE	Chang-GA	Chang-TS	Chang-SA
31	MPD(%)	0.510777	**0.0002**	0.0202	0.8973	0.1129
	MedPD(%)	0.000004	**0.000002**	1.1819	1.1992	1.2082
	Tiempo	223	**109**	621	469	476
85	MPD(%)	0.74099	**0.0052**	0.0136	3.5645	0.0394
	MedPD(%)	**0.00001**	0.0000211	0.0123	2.7816	0.0033
	Tiempo	**905**	1445	10332	9546	9412

comportamiento similar con las configuraciones que dieron un peor desempeño en las medidas de cantidad de puntos y el MPD. Para el problema sin restricciones, al probar diferentes configuraciones logramos identificar los mejores valores para los parámetros de configuración, sin embargo, hasta el momento aún no hemos realizado estas misma pruebas para el PSP con restricciones.

Para el problema formulado sin restricciones nuestro algoritmo produce soluciones de buena calidad que son competitivas con las heurísticas contra las que comparó el desempeño del algoritmo. Los resultado obtenidos para el PSP sin restricciones se presentan en la Tabla 2.

Establecimos la comparación de nuestro algoritmo BFO-PBIL mejorado contra PBILDE [16] y tres heurísticas propuestas en [7]. En [13] los autores emplearon medidas de desempeño diferentes por lo que no fue posible establecer una comparación con esta heurística. Con el objetivo de mostrar las mejoras que ofrece nuestra solución comparada con el algoritmo de BFO [13], se presenta en la Figura 2 las fronteras eficientes encontradas por las dos técnicas para el PSP sin restricciones. Como es posible observar, las mejoras introducidas al algoritmo permiten encontrar buenas soluciones ubicadas más cerca a la frontera eficiente real para los portafolios que ofrecen menor riesgo y menor retorno. Además, los

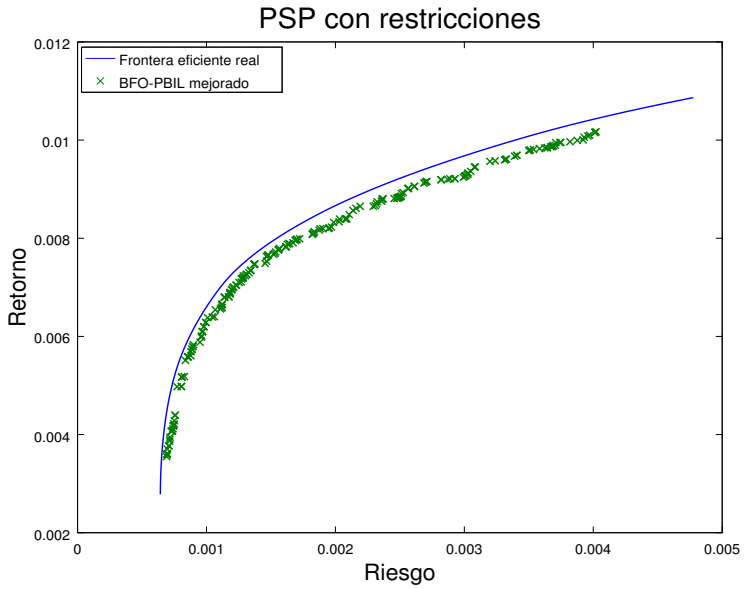


Fig. 1: BFO-PBIL mejorado para el problema (2). Conjunto de datos: 31 activos

portafolios encontrados por BFO-PBIL mejorado se aprecian mejor distribuidos sobre la frontera eficiente. Por otro lado, los dos algoritmos pudieron encontrar los mejores portafolios ubicados en el área de mayor riesgo y mayor retorno, para los cuales se observa una buena distribución sobre esta parte de la frontera eficiente.

5. Trabajo futuro

Los resultados aquí presentados son parte de un trabajo más amplio que aún se encuentra en curso. En dicho trabajo estamos analizando el algoritmo aquí propuesto con diferentes formulaciones del PSP y diferentes restricciones realistas que pocas veces son consideradas. En lo que respecta al algoritmo BFO-PBIL, es necesario probar los parámetros de configuración con distintos valores para el tamaño de la población N_B y el número de iteraciones de los pasos de eliminación-dispersión ($MAX_{Elim-Disp}$) y reproductivos (MAX_{Reprod}). Hemos visto que al utilizar el enfoque de PBIL es necesario aumentar el número de pasos reproductivos para dar tiempo al vector de obtener un aprendizaje significativo e incluirlo en las nuevas bacterias para llegar a buenos resultados. En este trabajo mostramos el potencial que tiene el algoritmo híbrido BFO-PBIL mejorado con una configuración estándar, sin embargo, es necesario realizar pruebas con conjuntos de datos más grandes, nuestro objetivo es proponer un algoritmo que sea robusto bajo un número grande de instancias como es el caso de los mercados bursátiles.

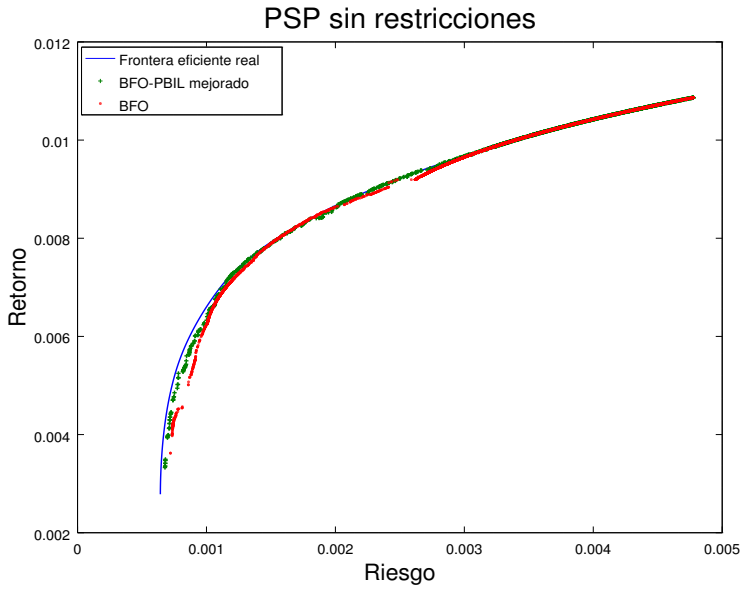


Fig. 2: Comparativa de BFO y BFO-PBIL mejorado para el problema (1). Conjunto de datos: 31 activos

6. Conclusiones

El algoritmo BFO es una de las heurísticas más novedosas en el área de Inteligencia de Enjambre. La técnica ha demostrado un gran potencial para resolver problemas de optimización en diferentes áreas y recientemente se ha empezado a utilizar para resolver el problema de la optimización de portafolios. En este trabajo hemos modificado el algoritmo original de BFO propuesto por Passino para mejorar algunas de las limitaciones que presentaba. Al incluir mejoras como una función lineal decreciente para el tamaño de los pasos quimiotácticos, reinicialización de las bacterias y asignación de pesos aleatorios durante el fase de reproducción hemos visto una mejora significativa en el desempeño del algoritmo. Además, hemos integrado y adaptado la técnica de aprendizaje incremental PBIL a BFO de manera exitosa agregando un componente que guía a las bacterias hacia buenas regiones con una adecuada exploración y explotación del espacio de búsqueda. Los resultados preliminares que hemos obtenidos hasta el momento mostraron que nuestro algoritmo es capaz encontrar soluciones de muy buena calidad que son competitivas con otros algoritmos de optimización.

Agradecimientos. El primer autor agradece el apoyo recibido por el CONACyT a través de una beca para estudios de posgrado.

Referencias

1. Markowitz, H.: Portfolio selection. The journal of finance, 1(7), 77–91 (1952)
2. Gupta, P., Mehlawat, M. K., Saxena, A.: Asset portfolio optimization using fuzzy mathematical programming. Information Sciences, 178(6), 1734–1755 (2008)
3. A. Ponsich, A.L. Jaimes, C.A.C. Coello: A Survey on Multiobjective Evolutionary Algorithms for the Solution of the Portfolio Optimization Problem and Other Finance and Economics Applications. IEEE Transactions on Evolutionary Computation, vol. 17, no.3, 321–344 (2013)
4. R. Ruiz-Torrubiano, A. Suarez, R. Moral-Escudero: Selection of optimal investment portfolios with cardinality constraints. In: Evolutionary computation. IEEE congress on CEC, pp. 2382–2388 (2006)
5. Vitoantonio Bevilacqua, Vincenzo Pacelli, Stefano Saladino: A novel multi objective genetic algorithm for the portfolio optimization. In: Advanced Intelligent Computing, Springer, pp. 186–193 (2012)
6. Hanhong Zhu, Yi Wang, Kesheng Wang, Yun Chen: Particle swarm optimization (pso) for the constrained portfolio optimization problem. Expert Systems with Applications, 38(8):10161–10169 (2011)
7. Chang, T.-J., Meade, N., Beasley, J.E., Sharaiha, Y.M.: Heuristics for cardinality constrained portfolio optimisation. Comp. & Opns. Res. 27, 1271–1302 (2000)
8. Doerner, K., Gutjahr, W., Hartl, R., Strauss, C., Stummer, C.: Pareto antcolony optimization: A metaheuristic approach to multiobjective portfolio selection. Annals of Operations Research, 131, 79–99 (2004)
9. Gaspero, L.D., Tollo, G., Roli, A., Schaerf, A.: Hybrid metaheuristics for portfolio selection problems. In: MIC 2007–Metaheuristics International Conference, Montreal (2007)
10. Passino, K. M.: Biomimicry of bacterial foraging for distributed optimization and control. IEEE Control Systems Magazine, 22, 52–67 (2002)
11. Tan, L., Niu, B., Wang, H., Huang, H., Duan, Q.: Bacterial foraging optimization with neighborhood learning for dynamic portfolio selection. Intelligent Computing in Bioinformatics, Springer International Publishing, pp. 413–423 (2014)
12. Niu, B., Xiao, H., Tan, L., Li, L., Rao, J.: Modified Bacterial Foraging Optimizer for Liquidity Risk Portfolio Optimization. Life System Modeling and Intelligent Computing, Springer Berlin Heidelberg, pp. 16–22 (2010)
13. Y. Kao, H.T. Cheng: Bacterial Foraging Optimization Approach to Portfolio Optimization. Computational Economics, vol. 42, num. 4, pp. 453–470 (2013)
14. S. G. Reid, K. M. Malan, A. P. Engelbrecht: Carry trade portfolio optimization using particle swarm optimization. In: 2014 IEEE Congress on Evolutionary Computation (CEC), pp. 3051–3058 (2014)
15. Beasley, J. E.: Or library dataset. (1999) http://people.brunel.ac.uk/ mastjjb/jeb/orlib/portinfo.html
16. K. Lwin, R. Qu: A hybrid algorithm for constrained portfolio selection problems. Applied intelligence, vol. 39, num. 2, pp. 251–266 (2013)

Control PI difuso de ganancias programables para un sistema mecatrónico de posicionamiento angular-lineal

Luis F. Cerecero-Natale[1,2], Julio C. Ramos-Fernández[2], Marco A. Márquez-Vera[2], Eduardo Campos-Mercado[2]

[1] Instituto Tecnológico de Cancún, Ingeniería en Mecatrónica, Cancún, Quintana Roo, México

[2] Universidad Politécnica de Pachuca,Posgrado en Mecatrónica, Zempoala, Hgo., México

lcerecero@itcancun.edu.mx, {luisfidelmeca, jramos, marquez, ecampos}@upp.edu.mx

Resumen. En este artículo se muestran los resultados experimentales de la integración sinérgica con: un sistema mecánico, un microcontrolador y una interface con el software EMC2 para aplicaciones de máquinas de control numérico (CNC). La contribución que se muestra en el presente trabajo, es el resultado experimental del diseño y programación de un controlador PI difuso de ganancias programables para el posicionamiento lineal de un sistema mecatrónico. La estrategia de diseño del controlador difuso es de 2-entradas y 2-salidas, el error y la derivada del error y las ganancias proporcional e integral, respectivamente. Se definieron 5 funciones de pertenencia del tipo gaussiana para fusificar el error y su derivada; se utiliza el conectivo AND en la premisa de las reglas difusas, para inferir el grado de disparo de cada regla difusa se codificó el operador producto (T-norm), el método del centroide es el mecanismo de defusificación. La estructura de la base de reglas son del tipoTakagi-Sugeno con los consecuentes de orden cero. Los resultados experimentales del control de posición en lazo cerrado, indican la viabilidad y efectividad de esta variante del controlador PI con ganancias programables para posicionamiento lineal de este tipo de servomecanismos. Este trabajo presenta los resultados experimentales comparativos, usando la regla clásica de sintonización de Ziegler-Nichols y el Controlador PI difuso de ganancias programables, para un tipo de sistema de posicionamiento, que es ampliamente utilizado en aplicaciones industriales.

Palabras clave: Controlador PI difuso, control de posición, máquinas herramientas.

Fuzzy Gain Scheduling PI Controller for Mechatronic Angle-Linear Positioning System

Abstract. In this paper we shows the experimental results using a microcontroller and hardware integration with the EMC2 software, using the Fuzzy Gain Scheduling PI Controller in a mechatronic prototype. The structure of the fuzzy

controller is composed by two-inputs and two-outputs, is a TITO system. The error control feedback and their derivative are the inputs, while the proportional and integral gains are the fuzzy controller outputs. Was defined five Gaussian membership functions for the fuzzy sets by each input, the product fuzzy logic operator (AND connective) and the centroid defuzzifier was used to infer the gains outputs. The structure of fuzzy rule base are type Sugeno, zero-order. The experimental result in closed-loop shows the viability end effectiveness of the position fuzzy controller strategy. To verify the robustness of this controller structure, two different experiments was making: undisturbed and disturbance both in closed-loop. This work presents comparative experimental results, using the Classical tune rule of Ziegler-Nichols and the Fuzzy Gain Scheduling PI Controller, for a mechatronic system widely used in various industries applications.

Keywords: Fuzzy gain scheduling, PI controller, position control, machine tools.

1. Introducción

El controlador PID es el más utilizado para controlar procesos industriales porque este tiene una simple estructura y un desempeño robusto. El diseño de este controlador solo necesita tres parámetros el proporcional, integral y derivativo, el controlador PID puede ser sintonizado usando la técnica bien conocida de Ziegler-Nichols [2]. Existen métodos alternativos para sintonizar las ganancias del PID que tienen estructuras similares a los controladores clásicos PID, donde las ganancias son adaptadas en línea basadas en auto-sintonización de redes artificiales wavenet para la estimación de parámetros [9]. Otra técnica muy utilizada es el control difuso (CD), con descripciones lingüísticas basadas en la experiencia de un experto humano, que se conoce como controlador del tipo Mamdani. Un trabajo de investigación que inspiró la presente propuesta, en donde se ilustra la filosofía y ejemplos para el diseño del controlador difuso PID de ganancias programables, se puede ver en [1].

En 1974 Mamdani fue uno de los pioneros [3], usando reglas de composición para inferencia que fueron propuestas por Zadeh [4], para controlar plantas no lineales. Un año después, Mamdani y Assilian desarrollaron el primer CLD, como se muestra en [5], este fue implementado satisfactoriamente para controlar un laboratorio tipo planta motriz de vapor. En un sentido estricto, el primer controlador difuso mostrado en [5] fue equivalente a un sistema difuso de 2 entradas PI, donde el error y la derivada del error, son usadas como entradas para el mecanismo de inferencia [6]. Takagi-Sugeno (TS) proponen una estructura de controladores difusos con los consecuentes en forma de polinomios [7], esta fue una propuesta que tendió un puente para el estudio y desarrollo de la lógica difusa integrando las técnicas de modelado y control clásico, moderno, adaptable y técnicas con descripción diferencial, donde los consecuentes de cada regla difusa están representados con estructuras matemáticas lineales o no lineales. Los controladores PID con diferentes configuraciones tipo TS se muestran con un análisis de estabilidad [8]. En la últimas tres décadas, en el campo de la aplicación de los controladores difusos, obtuvieron prestigio incluyendo aplicaciones: industriales, automotrices, aeroespaciales, médicas, agricultura de precisión, por citar algunas. Trabajos de investigación teórica en este sentido han aportado el soporte científico para

el diseño y desarrollo de controladores difusos, en donde se considera el estudio y análisis de estabilidad, como se ilustra en [15].

En el presente trabajo, la estructura del PI difuso de ganancias programables usa consecuentes lineales de orden cero (Singleton). Así, se muestran los resultados de aplicar el algoritmo de control PI difuso de ganancias programables, embebido en un microcontrolador, para controlar la posición angular-lineal de un sistema mecatrónico. Este artículo está organizado de la siguiente manera: en la Sección 2 se describe la estructura del Controlador PI Difuso de Ganancias Programables, la Sección 3 presenta los componentes de la Plataforma de Experimentación, la Sección 4 describe la Interfaz Hombre-Máquina del Sistema Mecatrónico, la Sección se 5 muestra el Diseño del controlador y los Resultados Experimentales. Las Conclusiones están incluidas en la Sección 6.

2. Controlador PI difuso de ganancias programables

En este trabajo se propone un Controlador PI Difuso de Ganancias Programables (CPIDGP), para controlar la posición Angular y en consecuencia el avance Lineal de un sistema mecatrónico, el bloque difuso consta de *2-entradas* y *2-salidas*, como se ilustra en la Fig. 1, las entradas son el error e y su derivada respecto del tiempo $\dot{e}$, las 2 salidas son las ganancias proporcional $K_{p_{Dif}}$ e integral $K_{i_{Dif}}$, ambas ganancias defusificadas se introducen al controlador PI clásico, de donde se obtiene la variable manipulada u, es decir la salida cambia cada instante dentro del ciclo de control.

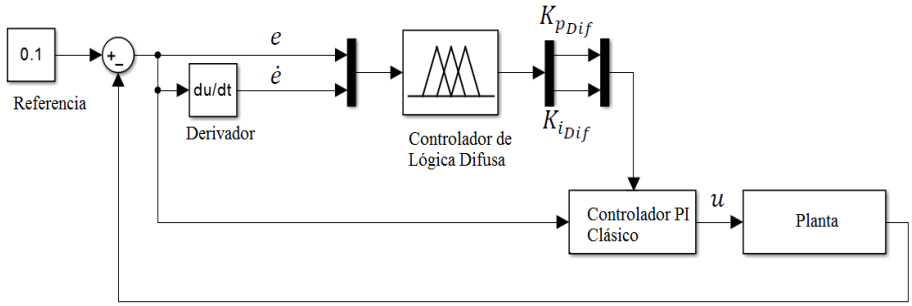


Fig.1. Sistema de Control tipo PI con Difuso de Ganancias Programables

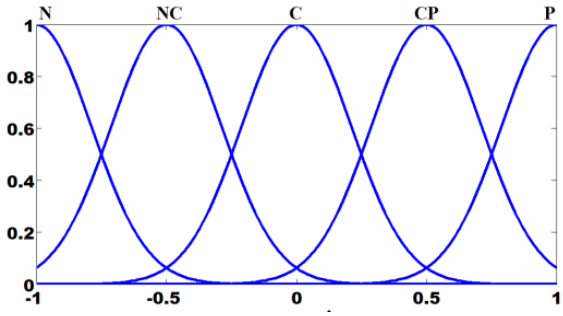


Fig. 2. 5 conjuntos difusas tipo Gaussianas para e y $\dot{e}$, normalizadas

Las variables lingüísticas e y $\dot{e}$, son definidas por *5* conjuntos difusos del tipo Gaussianas, mostradas en la Fig. 2, para estas $\sum_{i=1}^{5}\mu_i(e) = 1,\ \sum_{i=1}^{5}\mu_i(\dot{e}) = 1$, donde $\mu_i(e)$ y $\mu_i(\dot{e})$ son valores de las funciones de pertenencia en el instante dentro del ciclo de control.

Tabla 1. Reglas difusas de sintonización para las ganancias $K_{p_{Dif}}$ y $K_{i_{Dif}}$

Conjuntos difusos		**Salidas Singleton**	
error	**derivada del error**	**ganancias difusas**	
e	$\dot{e}$	$K_{p_{Dif}}$	$K_{i_{Dif}}$
N	N	1.00	1.00
N	NC	1.50	1.50
N	C	2.00	2.00
N	CP	2.50	2.10
N	P	3.00	2.20
NC	N	3.25	2.30
NC	NC	3.50	2.40
NC	C	3.75	2.50
NC	CP	4.00	2.60
NC	C	4.25	2.70
C	N	4.50	2.80
C	NC	4.75	2.90
C	C	5.00	3.00
C	CP	4.75	2.90
C	C	4.50	2.80
CP	N	4.25	2.70
CP	NC	4.00	2.60
CP	C	3.75	2.50
CP	CP	3.50	2.40
CP	C	3.25	2.30
P	N	3.00	2.20
P	NC	2.50	2.10
P	C	2.00	2.00
P	CP	1.50	1.50
P	C	1.00	1.00

El controlador difuso fue diseñado para calcular las salida $K_{p_{Dif}}$ y $K_{i_{Dif}}$ en cada ciclo de control. La base de conocimiento está compuesta por *25* reglas difusas, para los cinco conjuntos difusos, para las entradas del error y su derivada, normalmente son

las posibles combinaciones. En la Tabla 1 se muestra la base de reglas, que fue implementada en este trabajo.

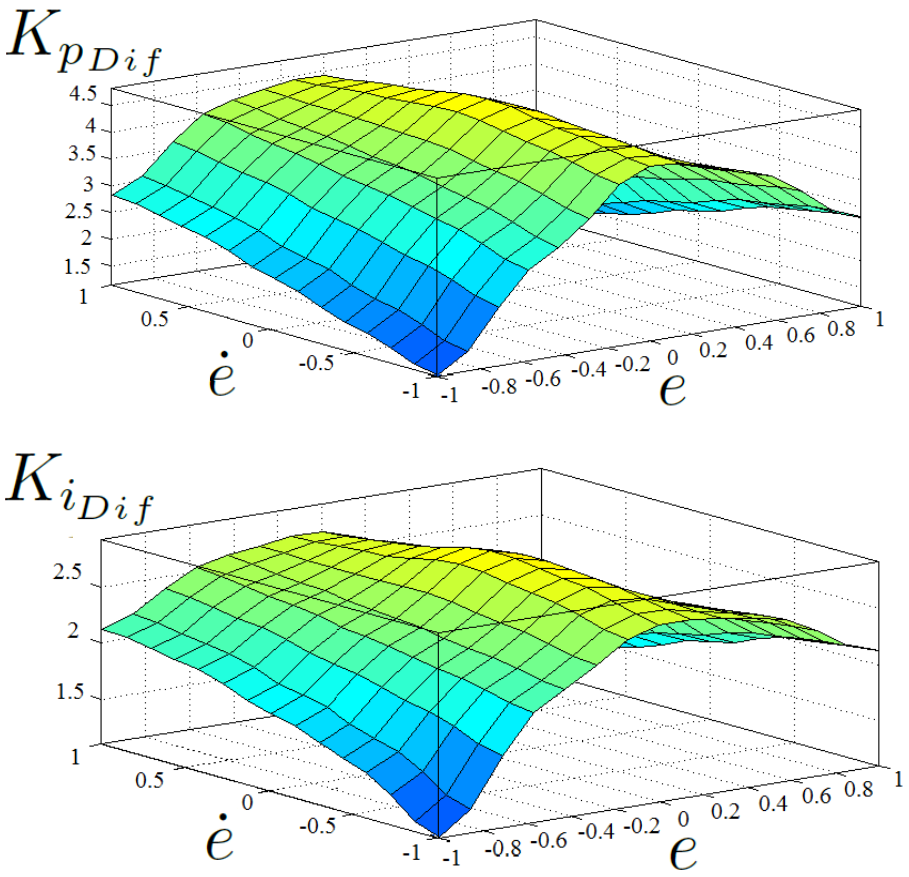


Fig. 3. Superficie de control de las salidas $K_{p_{Dif}}$ y $K_{i_{Dif}}$ del controlador difuso

La estructura de reglas difusas de CPIDGP se muestra en (1), método fusificación, defusificación y denormalización. La base de reglas difusas es:

$$R_j\text{: Si } e \text{ es } A_j \text{ and } \dot{e} \text{ es } B_j \text{ Entonces } K_{p_{Dif_j}} = a_j \text{ y } K_{i_{Dif_j}} = b_j, \qquad (1)$$

donde A_j y B_j son los conjuntos difusos dados por el diseñador, K_{pDif_j}y K_{iDif_j} son las ganancias proporcional e integral para $j = 1, 2, \dots, n$, donde n son las 25 reglas, a_j y b_j son los consecuentes con los valores de la Tabla I.

La ecuación (2) mecanismo de inferencia tipo producto:

$$\beta = (\mu_e \cdot \mu_{\dot{e}}). \qquad (2)$$

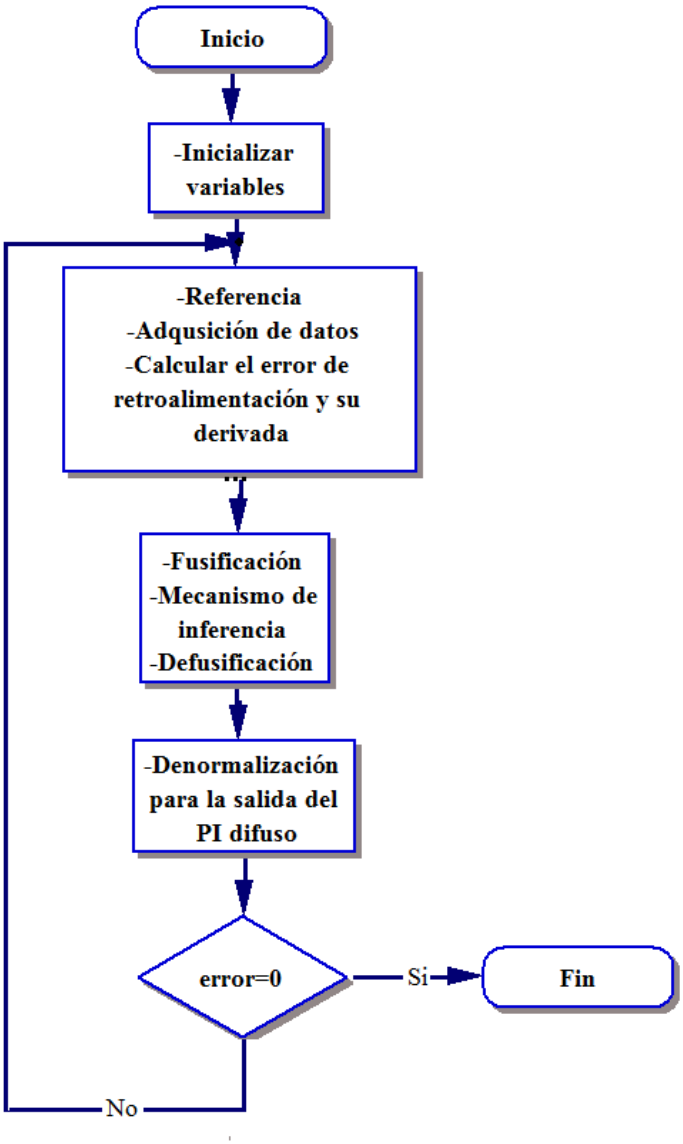


Fig. 4. Diagrama de flujo del controlador PI Difuso de Ganancias Programables

Defusificación de $K_{p_{Dif}}$y $K_{i_{Dif}}$ son las ecuaciones (3) y (4):

$$K_{p_{Dif}} = \frac{\sum_{j=1}^{n} \beta_j K_{p_{Dif}\,j}}{\sum_{j=1}^{n} \beta_j}, \tag{3}$$

$$K_{i_{Dif}} = \frac{\sum_{j=1}^{n} \beta_j K_{i_{Dif\,j}}}{\sum_{j=1}^{n} \beta_j}, \tag{4}$$

donde $K_{p_{Dif}} \in [1,\ 5]$ y $K_{i_{Dif}} \in [1, 3]$.

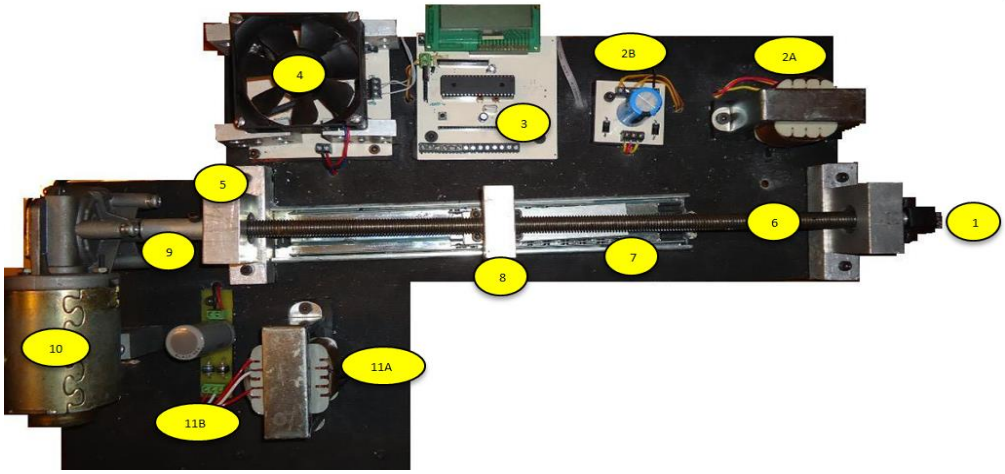


Fig.5. Componentes del sistema mecatrónico

El cálculo de cada ganancia, se realiza por medio de las ecuaciones (5) y (6):

$$K_p = K_{p_{Dif}} \cdot K_{pZ-N}, \tag{5}$$

$$K_i = K_{i_{Dif}} \cdot K_{iZ-N}, \tag{6}$$

donde K_{pZ-N} y K_{iZ-N} son ganancias calculadas a partir de la respuesta del sistema usando el método de sintonización de Ziegler–Nichols, en este caso, es necesario para obtener la respuesta al escalón del sistema mecatrónico de posicionamiento, la respuesta es de un sistema de primer orden con retardo en el transporte. Esta base de conocimiento es esencial en este artículo para el modelado e identificación del sistema. En este trabajo usamos por conveniencia las ecuaciones de transformación lineal (7) y (8) para normalizar e y $\dot{e}$ entre *-1* y *1*:

$$e' = \frac{e - e_{min}}{e_{max} - e_{min}}, \tag{7}$$

$$\dot{e}' = \frac{\dot{e} - \dot{e}_{min}}{\dot{e}_{max} - \dot{e}_{min}}. \tag{8}$$

La Fig. 3 muestra la superficie de control no lineal para las ganancias proporcional e integral difusas. En la Fig. 4 se muestra un diagrama de flujo del procedimiento que se ejecuta dentro del microcontrolador al compilar y ejecutar el CPIDGP.

3. Plataforma de experimentación

La Fig. 5 ilustra el prototipo donde fueron realizados los experimentos para verificar el desempeño del algoritmo CPIDGP propuesto en este artículo. La Tabla 2 contiene los nombres de los componentes que integran el sistema mecatrónico de posicionamiento Angular-Lineal.

Tabla 2. Componentes del sistema mecatrónico

(1)	Sensor de posición tipo encoder de 256 pulsos por revolución	(7)	Deslizador lineal
(2A, 2B)	Fuente de alimentación para sistema embebido (3)	(8)	Caja de tuerca
(3)	Sistema embebido de control basado en microcontrolador Microchip	(9)	Acoplamiento del Motor(10) con el husillo (6)
(4)	Puente H MOSFET, con sistema de ventilación por aire	(10)	Motor de Corriente Directa de Imán Permanente (PMDCM)
(5)	Sistema suspensión rígida (baleros)	(11A, 11B)	Fuente de alimentación de potencia, para (4) que controla a (10)
(6)	Husillo roscado,$\varphi = 12.7mm$, Resolución lineal $1.958mm = 1\,revolución$		

4. Interfaz hombre-máquina del sistema mecatrónico

El software Enhanced Machine Controller (EMC2) es mucho más que un simple programa de fresadora de Control Numérico Computarizado (CNC). Puede controlar máquinas-herramientas, robots u otros dispositivos automatizados. Se puede interactuar para controlar servomotores, motores paso a paso (stepper), relevadores y otros dispositivos relacionados con las máquinas herramientas. Hay cuatro componentes principales en el software EMC2: un controlador de movimiento, un controlador de Entradas/Salidas discretas, un ejecutor de tareas para coordinar las interfaces gráficas de usuario. Además hay una capa llamada HAL (Hardware Abstraction Layer), que permite la configuración de EMC2 sin la necesidad de compilar. La Fig. 6 es un diagrama de bloques que muestra la integración del sistema mecatrónico con la computadora y Linux como sistema operativo, donde se puede programar una trayectoria de posición por medio de Código G, para controlar el motor de corriente continua, la referencia de posición mediante el envío de señales a través del puerto paralelo. Normalmente, los actuadores son motores paso a paso que operan en lazo abierto, la contribución en el presente trabajo es el manejo de las operaciones de avance mecánico con un servomecanismo en lazo cerrado con un controlador del tipo difuso, la EMC2 también puede ejecutar a través de tarjetas de interfaz de servomotores o por medio de un puerto paralelo extendido para conectar con los tableros de control externo [10]. Una de las contribuciones de este desarrollo tecnológico es el control de posición

para máquinas-herramientas y se sometió a prueba el algoritmo embebido en el microcontrolador integrado: el Controlador PI Difuso de Ganancia Programable; el uso de la base de conocimientos mediante el método de sintonización a la respuesta al escalón de Ziegler-Nichols, para identificar los parámetros del Motor de Corriente Directa de Imán Permanente (MCDIP).

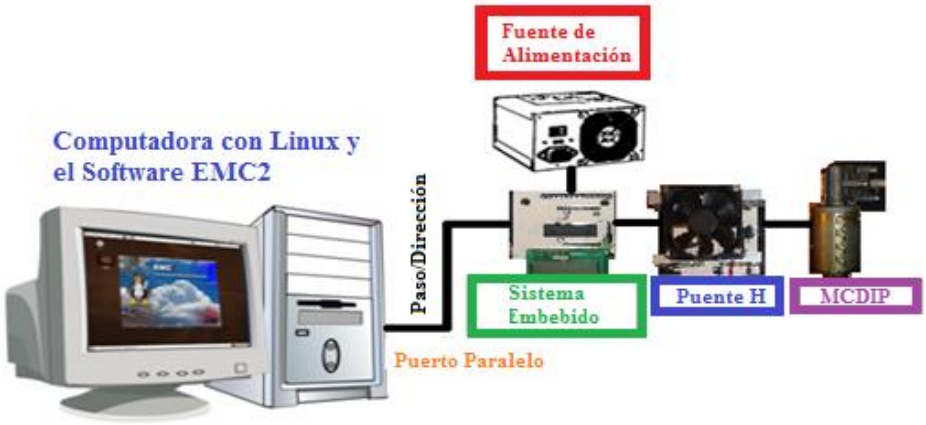


Fig.6. Interfaz Hombre-Máquina con el software EMC2

En este trabajo para la Interfaz Hombre-Máquina (HMI) se utilizó la EMC2, un pin del puerto paralelo con el fin de impulsar el sentido de giro y otro pin se utiliza para asignar la referencia, usando sólo 2 pines del puerto paralelo, por lo tanto puede controlar un sistema embebido, que consiste en el microcontrolador PIC18F4550 de Microchip, en el que se programó el algoritmo CPIDGP.

5. Resultados experimentales

Esta sección presenta la principal contribución de este artículo; la identificación de la planta fue realizada y esta descrita en [11]. Aquí, se desarrollan nuevos experimentos en la misma plataforma, donde la simulación numérica en lazo abierto de la función de transferencia en lazo abierto, se hace por medio del software MATLAB, posteriormente lo aplicamos en el microcontrolador en tiempo real. Presentamos dos clases de experimentos en tiempo real; uno sin perturbación (sin carga) y el segundo con perturbación (aplicando cargas externas).

5.1. Respuesta en lazo abierto

Para encontrar la función de transferencia experimental $\left(\frac{Posición}{\%\ PWM}\right)$de la plataforma, se aplica una señal constante del 80% del ciclo útil de trabajo de una señal Modulada en Ancho de Pulso (PWM) con una frecuencia de operación de $13\frac{rad}{s}$. De acuerdo al análisis de la gráfica de Bode como se muestra en [11], La función de transferencia es:

$$\frac{\theta(s)}{U(s)} = \frac{0.1076}{0.318s + 1} \cdot e^{-0.076s}, \tag{9}$$

donde $\theta(s)$ corresponde a la trasformación Angular-Linear de la posición que recorre la caja de tuerca (ver Fig. 5 and Tabla II el elemento (8)) y $U(s)$ es el ciclo útil de trabajo de la señal PWM. Al realizar la identificación paramétrica experimental se obtienen las siguientes constantes: ganancia del sistema 0.1076, la constante de tiempo de 0.318 y el retardo en el transporte 0.076, ambos en segundos.

5.2. Diseño del controlador difuso

El principal objetivo en el diseño del controlador difuso descrito en este artículo es regular y controlar la posición del sistema mecatrónico sin sobreimpulso en su posición. Porque en las máquinas herramientas de CNC, una respuesta dinámica con sobreimpulso es crítica y daña las piezas que se maquinan [12]. Usando el método de Ziegler-Nichols [13], se sintoniza el CPICZN, donde las ganancias proporcional e integral son: $K_{p_{Z-N}} = 3.7658$ y $K_{iZ-N} = 14.8650$, respectivamente. La Fig. 7 muestra los resultados de los controladores en una prueba de simulación, para contrastar las salidas del sistema usando el CPICZN contra el CPIDGP, donde la respuesta de tiempo del CPIDGP es más rápida que la del CPICZN, se observa que en la respuesta del control de posición ambas convergen a la referencia.

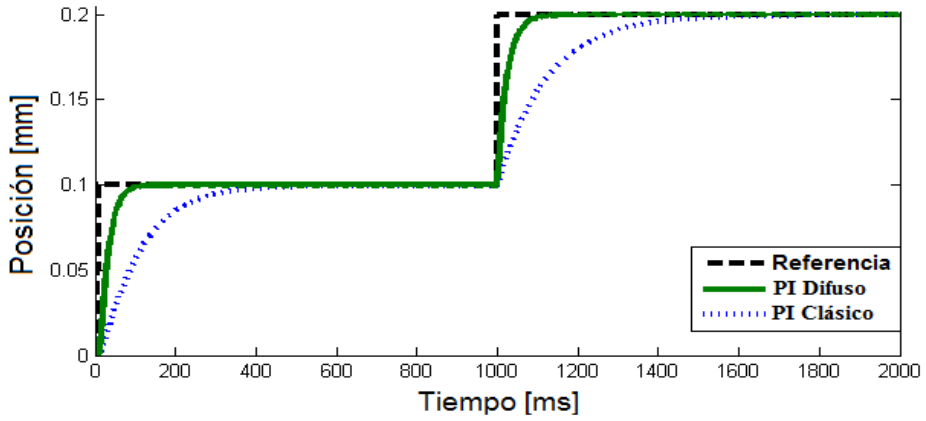


Fig. 7. Simulación del sistema en lazo cerrado bajo un cambio en la señal de referencia

5.3. Resultados experimentales sin perturbación

En esta Subsección se muestran los resultados experimentales de los controladores CPICZN y CPIDGP. Es importante enfatizar que los algoritmos de los controladores están corriendo en un simple microcontrolador PIC18F4550 de Microchip. La señal de salida PWM y de los pulsos del encoder son dados en [11], Donde cada pulso representa

$0.007632mm$ de avance lineal del mecanismo. Con el fin de seguir una trayectoria de referencia de $\frac{1}{10}mm$, es necesario que se acumulen *13* pulsos en el conteo del encoder.

La primera prueba experimental fue realizada sin carga (sin perturbación), el comportamiento de la plataforma con ambos controladores es mostrada en la Fig. 8, como se puede ver existe un retardo en ambos controladores, en el CPICZN son de $0.18s$ y 0.028s, para la primera y segunda referencia respectivamente, mientras que para el CPIDGP son de $0.116s$ y $0.104s$. Sin embargo el tiempo de convergencia del CPIDGP es más rápido que el CPICZN y sin sobreimpulso.

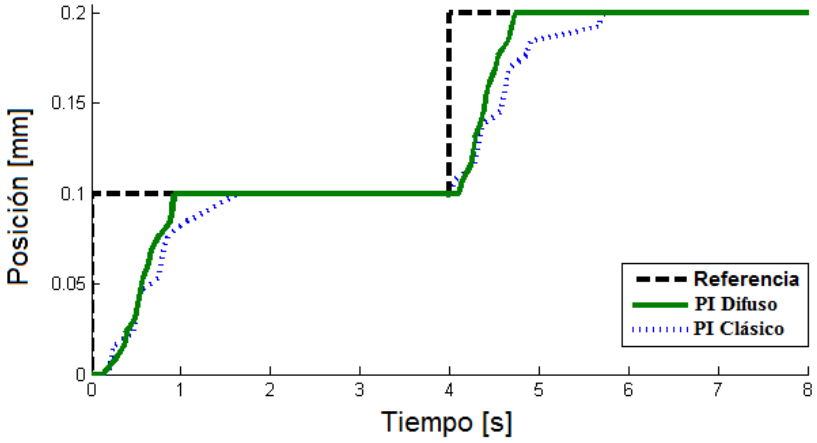


Fig.8. Desempeño de los controladores sin carga

5.4. Resultados experimentales con perturbación

La segunda prueba experimental se realiza con una carga de $2.788kg$, esta masa es colocada sobre la caja de tuerca. La respuesta dinámica con ambos controladores es mostrada en la Fig. 9 como se puede ver en esta figura, existe retardo en el transporte para ambos controladores. En el CPICZN son $0.212s$ y $0.06s$ para la primera y segunda referencia respectivamente. Mientras que para el CPIDGP son de $0.088s$ y $0.128s$. Sin embargo en el controlador CPICZN hay error en el estado estacionario, porque los parámetros de sintonización no son robustos.

La tercera prueba experimental se realiza con una carga de $4.934kg$ sobre la caja de tuerca, el comportamiento del CPICZN y CPIDGP son comparados en la Fig. 10, donde el retardo en el transporte para el CPICZN es: $0.212s$ y $0.264s$ para la primera y segunda referencia respectivamente, mientras que para el CPIDGP los retardos en la primera y segunda referencia son: $0.104s$ y $0.132s$. Al igual que el experimento anterior, el comportamiento del controlador CPIDGP en lazo cerrado, converge más rápido que el CPICZN, no reside error de retroalimentación en el estado estacionario y siempre llega a la señal de referencia deseada.

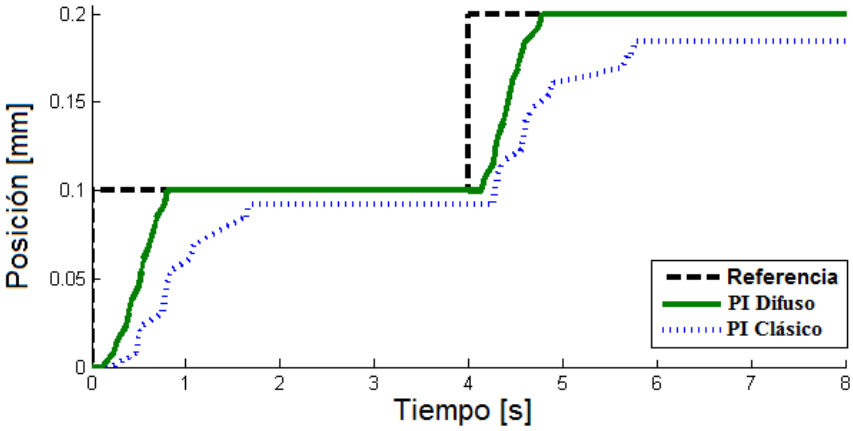


Fig. 9. Desempeño de los controladores con una carga de 2.788 kg

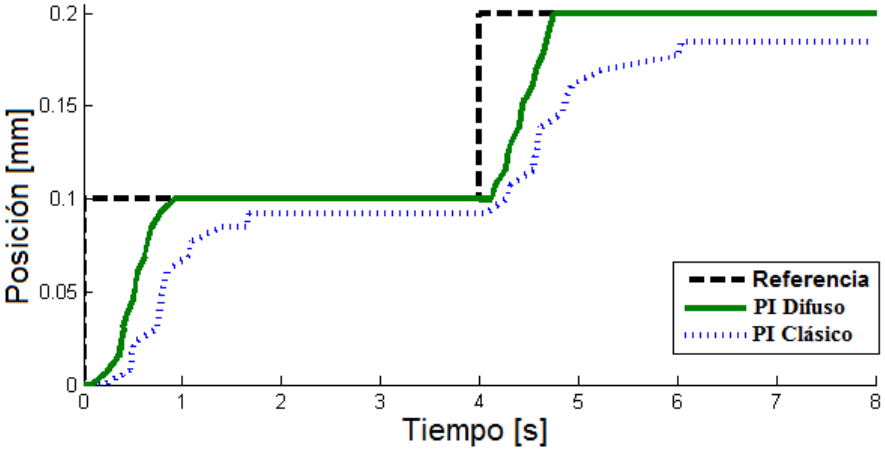


Fig. 10. Desempeño de los controladores con una carga de 4.934 kg

6. Conclusiones

La principal contribución en este artículo son los resultados experimentales usando el algoritmo embebido en un microcontrolador: CPIDGP con nulas respuestas de sobreimpulso y de error estacionario, estos efectos no se permiten en máquinas de desbaste o maquinados de control numérico. Estas dos características son importantes en los sistemas mecatrónicos, ya que puede ser utilizado en máquinas herramientas de alta precisión.

El algoritmo CPIDGP fue implementado en un microcontrolador de bajo costo 18F4550 de Microchip. Fueron utilizadas Funciones de membresía Gaussianas como entradas del CPIDGP, porque presentan una superficie más suave para inferir las

ganancias proporcional e integral difusas, sin sobreimpulso y con mínima oscilación en relación a la funciones de pertenencia triangular y trapezoidal. La principal contribución en este artículo es el uso de la interfaz EMC2 en lazo cerrado con el MCDIP, esto es una nota importante ya que los motores paso a paso son usados en la mayoría de las máquinas herramientas; sin embargo en este desarrollo tecnológico fueron reemplazados por MCDIP.

Referencias

1. Zhao, Z.-Y., Tomizuka, M., Isaka, S.: Fuzzy gain scheduling of PID controllers. IEEE Transactions on systems. Man and Cybernetics, Vol. 23, No. 5, pp.1392–1398 (1993)
2. Hang, C.C., Aström, K.J., Ho, W.K.: Refinements of the Ziegler-Nichols tuning formula. IEE Proc. D, Control Theory and Applications, Vol. 138, pp. 111–118 (1991)
3. Mamdani, E.H.: Application of fuzzy algorithms for control of simple dynamic plant. Proc. Inst. Elect. Eng. Contr. Sci., Vol. 121, pp. 1585–1588 (1974)
4. Zadeh, L.A.: Outline of a new approach to the analysis of complex systems and decision processes. IEEE Trans. Syst., Man, Cybernetic., Vol. SMC-3, No. 1, pp. 28–44 (1973)
5. Mamdani, E.H., Assilian, S.: An experiment in linguistic synthesis with a fuzzy logic controller. Int. J. Man-Mach. Stud., Vol. 7, No.1, pp. 1–13 (1975)
6. Mann, G.K.I., Hu, B.-G., Gosine, R.G.: Analysis of direct action fuzzy PID controller structures. IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics, Vol. 29, No. 3, pp. 371–388 (1999)
7. Takagi, T., Sugeno, M.: Fuzzy identification of system and its applications to modeling and control. IEEE Trans. Syst., Man Cybernetics, Vol. 15, No.1, pp. 116–132 (1985)
8. Ding, Y., Ying, H., Shao, S.: Typical Takagi-Sugeno PI and PD fuzzy controllers: analytical structures and stability analysis. Information Sciences, Vol. 151, pp. 245–262, (2003)
9. Gómez, O.I., Ramos-Velasco, L.E., Ramos-Fernández, J.C., García-Lamont, J., Espejel, M.A.: Implementation of different wavelets in an auto-tunig wavenet PID controller and its application to a DC motor. IEEE Electronics, Robotics and Automotive Mechanics Conference, pp. 301–306 (2011)
10. Organization Linux: Human-Machine Interface EMC2, http://www.linuxcnc.org/ [Accessed: 26 Oct. 2015]
11. Cerecero-Natale, L.F., Ramos Fernandez, J.C., Ramos-Velasco, L.E., Pedraza Vera, V.E.: Fuzzy gain scheduling PI controller for a mechatronic system. World Automation Congress (WAC), pp.1–5, Puerto Vallarta México (2012)
12. Overby, A.: CNC Machining Handbook, Building, Programming, and Implementation. McGraw-Hill (2011)
13. Aström, K.J., Wittenmark, B.: Adaptive Control. Addison-Wesley Series in Electrical and Computer Engineering: Control Engineering, (1989)
14. Aracil, J., Gordillo, F.: Stability Issues in Fuzzy Control. Physica-Verlag Heidelberg, Series Studies in Fuzziness and Soft Computing (2000)

Pronóstico del área de contacto de los neumáticos de un vehículo vía redes neuronales recurrentes.

Leopoldo Urbina, Marco A. Moreno-Armendáriz, Carlos A. Duchanoy, Hiram Calvo

Instituto Politécnico Nacional, Centro de Investigación en Computación, México

Resumen. En este trabajo, un sensor virtual predictivo basado en una red neuronal recurrente es propuesto para sensar el área comprendida entre los neumáticos y el terreno, la cuál es uno de los temas más importantes en la industria automotriz, de hecho, mantener una adecuada área de contacto del neumático garantiza la comodidad y la maniobrabilidad del vehículo. El sensor suave usa una red neuronal recurrente, el entrenamiento de una red neuronal recurrente es complicado para algoritmos basados en el cálculo del gradiente, debido a la aparición de algunos valles en la superficie de error. Como alternativa, el proceso de entrenamiento se puede modelar como un problema de optimización y este es resuelto mediante un algoritmo de evolución diferencial. Este sensor virtual ha sido validado con un banco de pruebas basado en el fenómeno de reflexión interna total frustrada.

Palabras clave: Redes neuronales recurrentes, sensor virtual predictivo, evolución diferencial.

Prediction of the Tire Contact Area of a Vehicle via Recurrent Neural Networks

Abstract. In this paper, a predictive soft sensor via a Recurrent Neural Network is proposed for sensing the area between the tires and the terrain, which is one of the most important issues in the automotive industry, in fact, keeping a tire contact patch suitable ensures the comfort and maneuverability of the vehicle. The core of the soft sensor is an recurrent neural network, the training of a recurrent neural network is complicated for algorithms based on the gradient, because of the apparition of some valleys on the error surface. As an alternative, the training process is interpreted like an optimization problem and it is solved by an differential evolution algorithm. This soft sensor has been validated using a test rig based on the frustrated total internal reflection phenomenon.

Keywords: Recurrent neural networks, predictive soft sensor, differential evolution algorithm.

1. Introducción

Los neumáticos son un componente importante para garantizar la seguridad de un vehículo terrestre. La razón radica en que el área de contacto entre la llanta y el suelo es donde las fuerzas del vehículo interactúan con el entorno, mantener un área de contacto adecuada garantiza el confort y la maniobrabilidad del vehículo. Un ejemplo de la importancia de monitorear el comportamiento de la llanta es la normativa impuesta en el 2005 por la Administración Nacional de Seguridad del Tráfico (NHTSA, por sus siglas en inglés), en la cual se exige la instalación de sistemas de monitoréo de presión del neumático (TPMS, por sus siglas en inglés) [1]. Además, en el estudio presentado en [6] se demuestra que las condiciones adversas del terreno aunado con los defectos en los neumáticos, son la mayor causa de accidentes en carretera. Los sensores en los neumáticos podrían ser usados como sistemas de prevención, también para mejorar la maniobrabilidad en condiciones de terreno desfavorables, además de ser parte de un sistema de control, como pueden ser las suspensiones adaptativas, sistemas de frenado anti bloqueo, sistemas avanzados de asistencia al conductor o prevención de coliciones. El desarrollo de los sensores del neumático presenta varios retos, los principales están reportados en [24]. El primer reto es desarrollar un proceso de medición directa para aplicaciones reales, el segundo es crear un sistema de comunicación inalámbrica entre la llanta y el vehículo que trabaje sin batería y el último es caracterizar y probar el sensor en condiciones reales.

En [21] se presentan algunas metodologías usadas para medición de distintos parámetros en al auto, como son presión y temperatura en los neumáticos, entre otros; del mismo modo se reportan metodologías para detección de fallas de un vehículo terrestre. Un primer intento al medir el área de contacto fue mediante la obtención de una imagen térmica de la llanta, en esta se podrán observar el área que está en contacto con el suelo, debido al calor generado por la fricción con el terreno, esta técnica presenta algunas desventajas, por ejemplo, es necesario que pase cierto tiempo para que la temperatura de la llanta aumente. Algunas alternativas para medir directamente el área de contacto son mediante un sensor piezoeléctrico, estas están reportadas en [20], [23], [25], [21] y [22]. Los trabajos previamente mencionados tienen algunos problema en común, debido a que los sensores se encuentran dentro del neumático, si el sensor se daña, se necesitan herramientas especializadas para el remplazo, además la manufactura requerida para el desarrollo del sensor es costosa. Una alternativa es obtener la medición de manera indirecta, por ejemplo, en [27] se usa un sistema de posicionamiento global (GPS, por sus siglas en inglés) para estimar los ángulos de deslizamiento longitudinal, del mismo modo en [4], se logra estimar la rigidez en los extremos del dibujo de una llanta mediante la información de velocidad en el GPS en conjunto con el sistema de navegación inercial (INS, por sus siglas en inglés).

Todos los sistemas físicos reales contienen parámetros que son difíciles de medir, ya sea que el costo del sensor sea elevado, el proceso de medición requiera de un largo tiempo o la medición requiere de un ambiente controlado. Con el fin de estimar las mediciones de dichas variables difíciles de medir, se desarrollaron los sensores virtuales. Un sensor virtual es un modelo computacional el cual permite la medición de casi cualquier parámetro en tiempo real [11]. El sensor virtual relaciona mediciones más sencillas de obtener con la medición de interés, todo a través de un modelo. Los sensores virtuales se dividen en, manejados por datos y manejados por modelo. Los sensores manejados por modelo, requieren forzosamente de un modelo matemático que describa el sistema. En la mayoría de los casos la dinámica del sistema es muy compleja para poderla modelar [29]. Los sensores virtuales manejados por datos no necesitan de un modelo matemático, estos sensores forman relaciones entre otras mediciones más fáciles de obtener y la medición deseada. Los sensores virtuales basados en datos requieren de un modelo de caja negra para simular el sistema, tales como, métodos estadísticos multivariantes, maquinas de soporte vectorial, redes neuronales. Por ejemplo, en [33], se propone un sensor virtual basado en el filtro de Kalman y modos deslizantes, para estimar el centro de gravedad y el ángulo de deslizamiento lateral del vehículo. En [26] se diseña un sensor virtual basado en una red neuronal con inversión izquierda, capaz de estimar los estados del vehículo tales como la velocidad de guiñada y el ángulo de deslizamiento lateral, estados que son importantes para el control de estabilidad del vehículo. En [7] se desarrolla un sensor virtual mediante la aplicación de un filtro de Kalman extendido, para estimar el coeficiente de fricción y el ángulo de deslizamiento lateral.

Las redes neuronales son usados comúnmente en los sensores virtuales basados en datos debido a sus capacidades como aproximadores universales [17]. Dependiendo la arquitectura y el tipo de conexiones que tenga una red neuronal, estas pueden ser aplicada para resolver problemas con dinámica compleja, una red neuronal recurrente (RNN, por sus siglas en inglés) puede resolver problemas modelados como series de tiempo. Por lo tanto un sensor virtual que use una red neuronal recurrente, además de poder estimar mediciones complejas de un sistema dinámico, es capaz de predecirlas. Un ejemplo de sensor predictivo se presenta en [12], donde se usa un modelo de inferencia neuro-difuso adaptativo para predecir el comportamiento de un vehículo en la maniobra de cambio de carril. En [31] se presenta un sistema de control inteligente de neumático, sensores piezoeléctricos se usan como entrada a una red neuronal la cual estima y predice, las diferentes fuerzas presentadas en el neumático.

Una de las principales problemáticas en el uso de redes neuronales es el entrenamiento. El proceso de entrenamiento consiste en ajustar los pesos sinápticos y bías de la red hasta que todos los conjuntos de entrada de la red estén relacionados con los valores deseados a la salida. Existen distintos métodos con los cuales la red puede ser entrenada, la mayoría de estos métodos tratan de calcular el gradiente o en el mejor caso, aproximarlo, el algoritmo de Levenberg-Marquardt [30] es uno de ellos. Estos métodos son eficientes para el entrenamiento de redes neuronales sin retroalimentación, en el caso de las redes recurrentes existen

diversas dificultades [2]. La principal razón es la presencia de valles espurios en la superficie del error; estos valles son mínimos locales y cambian dependiendo de la secuencia de entrada [8]. En [28] se describe una forma de lograr un entrenamiento exitoso sin caer en un valle espurio, basta con monitorear la magnitud del gradiente, si este es muy grande, es un indicio de que estamos en un valle, entonces el conjunto de entrada se cambia temporalmente para entrenar con otra secuencia. Existen otros métodos para lograr un entrenamiento exitoso, en [3] sugieren usar algoritmos de búsqueda estocástica, tales como el algoritmo de recocido simulado, evolución diferencial, entre otros. El problema de entrenamiento puede ser interpretado como un problema de optimización, por lo tanto puede ser resuelto por métodos no basados en el gradiente [15]. En [19] es usado un algoritmo de evolución diferencial como base de entrenamiento de una red neuronal, estos métodos estocásticos pueden evitar perderse en los valles espurios debido a su mecanismo de búsqueda.

En este trabajo se propone una aplicación de una red neuronal recurrente en un sensor virtual predictivo. Este sensor sería útil en un sistema de suspensión activa ya que el sistema podría predecir perturbaciones. Además se propone un arreglo de sensores que nos permita medir un conjunto de variables que están relacionados con el área de contacto del neumático.

2. Presentación del problema

El área de contacto entre el neumático y el terreno es el último y el más importante eslabón en la cadena del sistema de propulsión, debido a que es dónde todas las fuerzas externas interactúan con el vehículo. Uno de los pasos más importantes en el desarrollo de un sensor virtual es elegir el conjunto de variables físicas que estén estrechamente relacionadas con el área de contacto de la llanta, la razón radica en que una vez elegidas, se elaborará toda la instrumentación electrónica para sensar estas variables. A pesar del desarrollo en el campo de los sensores virtuales, no se ha encontrado trabajo alguno en el cual se use un sensor virtual para estimar el área de contacto de la llanta. Existen distintas metodologías para obtener de manera directa esta área, en [14] se estima el área de contacto por medio de figuras geométricas, por otra parte, en [32] se expone un modelo de elemento finito para calcular la misma área. El vehículo en el que el sensor virtual será probado es el Baja 5sc SS de hpi-racing® [18]. El Baja 5sc SS es un vehículo de radio control a escala 1/5, fabricado con un motor de 2.9 HP de dos tiempos, este vehículo posee todos los sistemas de un auto real. Se escogió este vehículo debido a al facilidad con la que se puede instrumentar e implementar los sensores, ademas gracias a las dimensiones del vehículo, este puede ser sometido a pruebas en el banco de pruebas desarrollado, por último, este vehículo no requiere de equipo especial para realizar ciertos ajustes.

3. Diseño del sensor predictivo

En [11], Fortuna propone 4 pasos para el diseño general de un sensor virtual, en este trabajo se ajustaron estos pasos para nuestra aplicación.

3.1. Selección de variables

El primer paso en el diseño de un sensor virtual es el análisis de los datos disponibles, el resultado de este análisis es una lista de variables que estén estrechamente relacionadas con el área de contacto del neumático. A partir de esta selección, se diseñará la instrumentación necesaria. Como apoyo, se usó un modelo matemático previamente desarrollado por nuestro grupo en [9]. El modelo matemático fue desarrollado para un vehículo distinto al propuesto, de cualquier modo el modelo nos ayudó a reconocer las variables clave que afectan en mayor medida al área de contacto. El resultado del estudio fue el siguiente conjunto de variables, la aceleración de las llantas en el eje z (a_t_1, a_t_2, a_t_3, a_t_4), el desplazamiento de la suspensión en cada amortiguador (x_s_1, x_s_2, x_s_3, x_s_4), el ángulo de cabeceo y balanceo lateral del chasis ((Φ y Θ respectivamente) y por último el ángulo de la dirección (Ω). Para futuras referencias el número de cada llanta empieza de la frontal izquierda y continua en contra del sentido de las manecillas del reloj.

3.2. Instrumentación y adquisición de datos

La instrumentación electrónica debe poder ser capaz de sensar las variables físicas que se requieren, a pesar de las vibraciones del terreno o del motor. El primer conjunto de mediciones son los desplazamientos de la suspensión en cada amortiguador. Los sensores escogidos fueron los sensores de flexión de SpectraSymbol®. La razón por la que se eligió este sensor es que no interfiere en el movimiento natural de la suspensión ya que la fuerza necesaria para doblar el sensor es despreciable comparado con la fuerza en el amortiguador. El sensor se coloca entre el chasis y la llanta, a medida que el sensor se deforma, este cambiará su resistencia interna, por lo tanto estos sensores requieren de un circuito que transforme los cambios de desplazamiento en diferencia de potencial. El segundo conjunto de mediciones son las aceleraciones. Se seleccionó el acelerómetro ADXL345 de Sparkfun® debido a que cumplía los requerimientos de la instrumentación. Para leer el acelerómetro solo es necesario orientar el sensor a los ejes del sistema y establecer un protocolo de comunicación, normalmente este es establecido por medio de un microcontrolador. Los ángulos de cabeceo y balanceo lateral del chasis pueden ser leídos por un giroscopio, pero se eligió una unidad de masa inercial (IMU, por sus siglas en inglés), una IMU puede sensar aceleración, orientación y posición angular por cada eje. La IMU que cumple los requerimientos de la instrumentación es la Razor IMU de Sparkfun®, al igual que el acelerómetro, antes de empezar a sensar, es necesario colocar la IMU orientada con los ejes del chasis. La última variable es el ángulo de la dirección. El proceso del vehículo para realizar un cambio en la dirección es el siguiente,

el piloto indica el cambio usando un transmisor de radio frecuencia, el vehículo recibe la señal mediante un receptor de radio frecuencia y es convertida en una señal de modulación por ancho de pulsos (PWM, por sus siglas en inglés), el PWM modifica la posición de un servomotor y este a su vez por medio de la barra de dirección modifica el ángulo de la dirección. El punto en el que se puede sensar el ángulo sin alterar o modificar este proceso es copiando la señal de PWM por medio de un contador o timer.

Para finalizar la instrumentación se requiere de un microcontrolador que lea y administre los datos de los sensores. Se eligió el microcontrolador Tiva TM4C123GXL de Texas Instruments® debido a que cuenta con los suficientes módulos para interpretar la señal de los sensores de flexión, del mismo modo puede codificarse el protocolo de comunicación entre los acelerómetros y la IMU además tiene un módulo para capturar la señal PWM del receptor. El principal reto de la instrumentación es asegurar que todas las mediciones sean tomadas en el mismo tiempo, o con un retraso mínimo.

3.3. Medición del área de contacto del neumático

El banco de pruebas para medir el área de contacto de las llantas está basado en el fenómeno de reflexión interna total, el cual ocurre cuando la luz incide sobre un medio con menor índice de refracción (del vidrio al aire), el rayo se desvía de la normal y para cualquier ángulo de incidencia menor que el ángulo crítico, parte de la luz incidente será transmitida y parte será reflejada. Ver [10] para más información sobre el fenómeno. La tecnología basada en la reflexión interna total frustrada (FTIR, por sus siglas en inglés), añade una superficie extra para interactuar con el medio óptico. El banco de pruebas basado en la tecnología FTIR consta de un panel de acrílico con leds infrarrojos alrededor de su periferia, los leds administran la luz infrarroja dentro del panel. Cuando una neumático toca el panel de acrílico, debido al mayor indice de refracción causado por el neumático, la luz infrarroja es reflejada del área que está en contacto con el panel. Debajo del panel, el banco de pruebas tiene instalada una cámara para capturar los rayos infrarrojos reflejados. La imagen tomada por la cámara pasa por un filtro pasa bandas, con el fin de solo quedarnos con el espectro infrarrojo, que es el área en contacto con el panel; después la imagen pasa por un proceso de agrupación y conteo, el cual consiste en separar las áreas de cada llanta y contar los pixeles de cada área. De este modo se obtiene la medida del área de contacto de la llanta.

3.4. Red neuronal recurrente como parte del sensor virtual

Existen distintos tipos de métodos para aproximar un modelo de caja negra, algunos de ellos son; maquinas de soporte vectorial, métodos estadísticos multivariantes, métodos adaptativos, redes neuronales y métodos híbridos. En este trabajo se eligió el modelo de redes neuronales recurrentes debido al gran desempeño que han tenido modelando sistemas dinámicos complejos, además del desempeño en la predicción de diversos sistemas [5]. La capacidad de predicción

de una red neuronal recurrente es debido a sus conexiones de retroalimentación, estas conexiones permiten a la RNN aprender series de secuencias. La arquitectura más usada de una RNN consta de un perceptrón multicapa con ciclos de retroalimentación. Para el diseño del sensor virtual se propone una red neuronal no lineal autorregresiva con entrada exógena (NARX, por sus siglas en inglés). La arquitectura de la NARX es la siguiente, necesitamos 11 entradas (variables elegidas) más un bloque de retraso de 3 tiempos, 3 entradas de retroalimentación de la salida (las 4 áreas de contacto), esto nos da un total de 56 entradas que van están conectadas a una capa oculta de 20 neuronas; la capa de salida consta de las 20 entradas de la capa anterior, más 3 entradas de retroalimentación de la salida (las 4 áreas de contacto), dando un total de 32 entradas que se conectan finalmente a 4 neuronas.

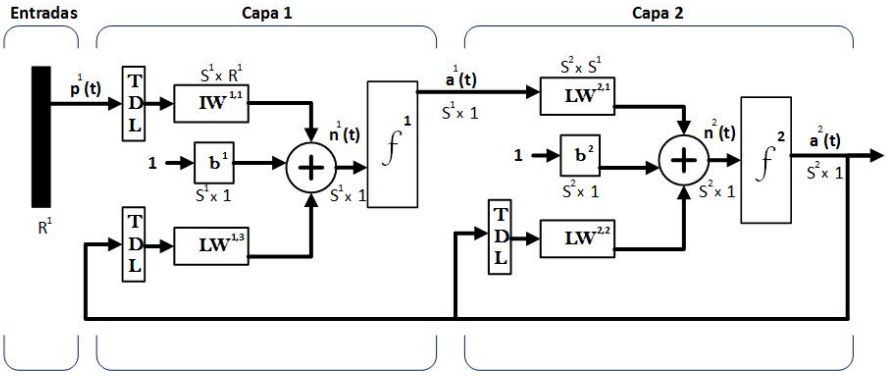


Fig. 1. Red neuronal NARX

Esta tipo de red puede ser entrenada usando algoritmos basados en el cálculo del gradiente (Ver [13]). Estos métodos presentan dificultades debido a la presencia de valles espurios en la superficie del error (Ver [16]). Estos valles son ocasionados por el conjunto de entrada y no tienen relación alguna con el mínimo real. Este problema puede ser considerado como un proceso de optimización, dónde los valles espurios pueden ser interpretados como mínimos locales. Dado este planteamiento, los métodos de optimización estocástica pueden presentar una gran ventaja respecto a los métodos basados en el gradiente, esto es debido al mecanismo de exploración que los métodos estocásticos presentan, esto les permite sortear los mínimos locales y no perderse en ellos. En este trabajo, seleccionamos como método de entrenamiento una versión modificada del algoritmo de entrenamiento basado en evolución diferencial que fue propuesto inicialmente para redes con propagación hacia adelante en [19], el cual se aplicó a una red neuronal recurrente.

Entrenamiento con el algoritmo de evolución diferencial El algoritmo de evolución diferencial puede ser clasificado como un algoritmo decodificador evolutivo ideal para espacios de búsqueda continua. Por esta razón, este puede ser aplicado para optimizar los pesos y bías de una red neuronal recurrente. La salida de la red neuronal recurrente está en función a los pesos y bías de cada capa k. Ver ecuación 1.

$$L(k) = f(w_{K,i,j,Q} * Int_K + b_{K,i,Q}), \tag{1}$$

donde $w_{K,i,j,Q}$ es la matriz de pesos, Int_K es el vector de entrada, $b_{K,i,Q}$ es el vector de bías, $f(*)$ es la función de transferencia y $L(k)$ es la salida de la capa.

En el entrenamiento clásico, las entradas de la primera capa de la red y la salida son datos conocidos, los pesos y bías de la red son ajustados con el fin de obtener un mapeo del conjunto de entrada con el de salida. El ajuste de pesos y bías se lleva a cabo reduciendo al mínimo la función de error de la red (ecuación 2).

$$E = \frac{1}{2}(L(F) - T_g)^2, \tag{2}$$

donde T_g es el salida deseada y $L(F)$ es la salida actual de la red.

De este modo, el problema de entrenamiento es ahora un problema de optimización en el que el objetivo es minimizar la función de error al ajustar los valores de pesos y bías de la red. Como la mayoría de algoritmos evolutivos, el algoritmo de evolución diferencial propone una población de candidatos a solución, no solo una única solución. El algoritmo de evolución diferencial usa un esquema de reproducción de la población diferente de otros algoritmos evolutivos. Después de la inicialización de la primera población, los vectores de población de las nuevas generaciones son tomados al azar y combinados para crear nuevos vectores mutados de pesos y bías (ecuación 3 y 4), a este paso se le conoce como mutación.

$$u_{K,i,j} = w_{K,i,j,R1} - F * (w_{K,i,j,R2} - w_{K,i,j,R3}), \tag{3}$$

$$v_{K,i,j} = b_{K,i,R1} - F * (b_{K,i,R2} - b_{K,i,R3}), \tag{4}$$

donde F es la constante de mutación.

Los vectores mutados son combinados con el vector original usando un factor de cruza CR, el cual generará un nuevo candidato, a esta etapa se le conoce como mutación. Finalmente, el desempeño del individuo original y el nuevo candidato a solución son comparados y se elimina al que presente el peor desempeño, quedándose el mejor individuo para la siguiente generación (Ver algoritmo 1).

4. Experimentos y resultados

4.1. Condiciones del experimento

El banco de pruebas construido permite una medición precisa del área de contacto del neumático. El principal inconveniente es que no permite realizar

Algorithm 1 Algoritmo de entrenamiento basado en evolución diferencial

Mientras La condición de termino no sea alcanzada **Hacer**
  Para Cada elemento Q de la población **Hacer**
    Seleccionar Números enteros aleatorios $R1, R2, R3 \in [1, Población]$
    Para Cada capa K de la red neuronal **Hacer**
      Mutar cada i, j peso de la red
      $u_{K,i,j} = w_{K,i,j,R1} - F * (w_{K,i,j,R2} - w_{K,i,j,R3})$
      Mutar cada i bías
      $v_{K,i,j} = b_{K,i,R1} - F * (b_{K,i,R2} - b_{K,i,R3})$
      Cruzar cada i, j peso de la red
      Si $Rand(0, 1) \leq CR$
        $wnew_{K,i,j,Q} = u_{K,i,j}$
      De otro modo
        $wnew_{K,i,j,Q} = w_{K,i,j,Q}$
      Fin Si
      Cruzar cada i Bías
      Si $Rand(0, 1) \leq CR$
        $bnew_{K,i,Q} = v_{K,i,Q}$
      De otro modo
        $bnew_{K,i,Q} = b_{K,i,Q}$
      Fin Si
    Fin Para
    Para Cada secuencia de entrenamiento **Hacer**
      Para Cada capa K de la red neuronal
        Evaluar la RNN usando los valores anteriores de pesos y bías
        $Lo_k = f(w_{K,i,j,Q} * Int_K + b_{K,i,Q})$
        $fitO_k = \frac{1}{2}(Lo - T_g)^2$
        Evaluar la RNN usando los valores nuevos de pesos y bías
        $Ln_k = f(wnew_{K,i,j,Q} * Int_K + bnew_{K,i,Q})$
        $fitN_k = \frac{1}{2}(Ln - T_g)^2$
      Fin Para
      Seleccionar las soluciones y compararlas con el objetivo
      Si $sum(fitO_k) \geq sum(fitN_k)$
        $b_{K,i,Q} = bnew_{K,i,Q}$
        $w_{K,i,j,Q} = wnew_{K,i,j,Q}$
      De otro modo
        $b_{K,i,Q} = b_{K,i,Q}$
        $w_{K,i,j,Q} = w_{K,i,j,Q}$
      Fin Si
    Fin Para
  Fin Para
Fin Mientras

mediciones del auto en movimiento. Por lo tanto, se desarrollaron una sucesión de movimientos que simulan una pista de prueba, esta sucesión contiene maniobras de giro, lapsos de aceleración y frenado.

4.2. Entrenamiento de la red neuronal recurrente

El entrenamiento de nuestra RNN consiste en ajustar los parámetros de la RNN (pesos y bías) por medio del algoritmo de evolución diferencial expuesto en la sección 3.4. El entrenamiento se considera exitoso si el conjunto de variables de entrada mapea al valor de salida deseado. La precisión del sensor virtual aumentará, si el conjunto de entrenamiento contiene todos los movimientos posibles en el vehículo. Por esta razón es primordial simular y sensar todos estos movimientos en el banco de pruebas.

La arquitectura de la red neuronal recurrente NARX es la siguiente, 11 entradas (variables elegidas) más un bloque de retraso de 3 tiempos (total de 44 entradas), más 3 entradas de retroalimentación de la salida (las 4 áreas de contacto), esto nos da un total de 56 entradas que van están conectadas a una capa oculta de 20 neuronas ($S^1 = 20$); la capa de salida consta de las 20 entradas de la capa anterior, más 3 entradas de retroalimentación de la salida (las 4 áreas de contacto), dando un total de 32 entradas que se conectan finalmente a 4 neuronas($S^2 = 4$). Para el proceso de entrenamiento se realizaron 4 experimentos cada uno con distintas condiciones de manejo y terreno, cada uno de estos experimentos contiene un total de mil datos de medición. El horizonte de predicción propuesto en este trabajo es de $33ms$, este tiempo es mayor al tiempo de respuesta de una suspensión activa. El horizonte de predicción elegido es la diferencia entre anticiparse a una perturbación o reaccionar a ella. Para el algoritmo de evolución diferencial se uso un factor de cruza $CR = 0.5$, un valor de constante de mutación $F = 0.5$. Bajo estos criterios el entrenamiento se realizó de manera exitosa. Una vez ajustados los parámetros de la RNN esta puede ser embebida en cualquier hardware capaz de recibir las señales de los sensores para así predecir el área de contacto.

4.3. Resultados y comparaciones

Una vez entrenada la RNN, con el objetivo de evaluar el desempeño del sensor virtual predictivo, se realizó una secuencia de movimientos de prueba, estos movimientos fueron sensados tanto por la instrumentación como por el banco de pruebas, el objetivo de la prueba es comparar los resultados obtenidos por la RNN y el banco de pruebas. Cada experimento de prueba consta de mil datos de medición tanto del banco de pruebas como del sensor virtual. En la figura 2 se pueden observar los comparación entre las mediciones del sensor virtual y las del banco de pruebas. Dónde la linea punteada es la medición obtenida por el sensor virtual y la linea continua es la medida obtenida por el banco de pruebas.

Se puede notar que la señal predicha es casi idéntica a la medida real, la medida predicha presenta un desfase de $33.33ms$ que es el horizonte de predicción que se había escogido previamente, la tabla 1 contiene los resultados promedio de las pruebas realizadas a cada llanta. De manera general, el error obtenido entre el sensor virtual y el banco de pruebas es aceptable dado que en promedio, el porcentaje de error es menor al 10 % que es aproximadamente $75mm^2$.

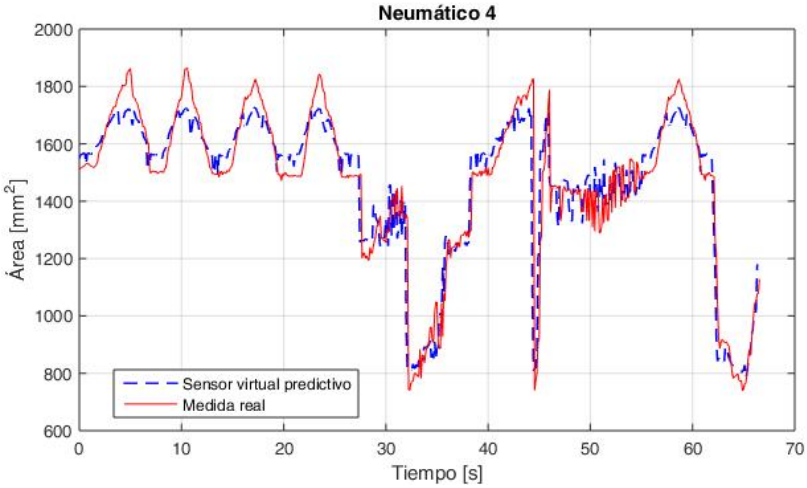


Fig. 2. Comparación entre las mediadas del banco de pruebas y el sensor virtual de la llanta 4

Tabla 1. Experimentos de prueba

Promedios	**Llanta 1**	**Llanta 2**	**Llanta 3**	**Llanta 4**
Promedio del error de predicción (mm^2)	22.32	7.25	32.28	25.36
Porcentaje promedio del error de predicción (%)	9.0997	5.8069	5.4906	4.0049

4.4. Discusión

Elegir un horizonte de predicción influye en la precisión de la predicción. Si se elige un horizonte de predicción muy amplio, la predicción es más susceptible a fallar. Por otro lado, pensando en una aplicación real, nuestro problema está acotado al tiempo de respuesta del amortiguador magnetoreológico en una suspensión activa, que para nuestro caso es menor a $33.33ms$. El horizonte de predicción propuesto permitirá el correcto funcionamiento de la suspensión activa. En la figura 2 se puede apreciar que el área predicha por el sensor virtual es muy similar al área medida una vez alcanzado el horizonte de predicción. Esto demuestra que el horizonte escogido es funcional, ya que predice de manera correcta además de permitir el correcto funcionamiento de una suspensión activa. Los resultados de la tabla 1 indican que el porcentaje promedio del error es de 9.0997 %. El sensor virtual puede predecir el área de contacto de los neumáticos de manera precisa.

5. Conclusiones

Este trabajo realizó un sensor virtual predictivo mediante redes neuronales recurrentes capaz de predecir el área de contacto de los neumáticos $33.33ms$ en el futuro. Este tiempo permite una predicción precisa del área, así como la activación de una suspensión activa, esto con el fin de anticiparse a las perturbaciones, en lugar de reaccionar a ellas. El problema de entrenar una red neuronal recurrente por métodos basados en el cálculo del gradiente se debe a la presencia de valles espurios. Debido a la capacidad de búsqueda de los algoritmos estocásticos, estos son una excelente alternativa para el problema. Se realizó la instrumentación electrónica necesaria para sensar las variables físicas seleccionadas, de igual modo se diseñó y construyó un banco de pruebas capaz de medir el área de contacto de las llantas. Por último se probó la eficiencia del sensor virtual predictivo demostrando tener una porcentaje de error promedio menor al 10 %

6. Trabajo a futuro

Con el deseo de continuar con el desarrollo alcanzado por este trabajo, se proponen las siguientes ideas: Seleccionar el menor número de variables de entrada pero que a la vez tengan una mayor relación con el área de contacto de la llanta, usar el sensor virtual como parte de un sistema de suspensión activa, finalmente, llevar la metodología a un vehículo convencional.

Agradecimientos. Los autores agradecen a los revisores por sus sugerencias, las cuales ayudaron a mejorar la calidad de la investigación, del mismo modo agradecemos el apoyo del Instituto Politécnico Nacional (SIP-IPN, COFAA-IPN, BEIFI-IPN) y del gobierno mexicano (SNI y CONACYT).

Referencias

1. Administration, N.H.T.S., et al.: Federal motor vehicle safety standards; tire pressure monitoring systems; controls and displays. Tech. rep., Technical report, Department of Transportation, http://www. nhtsa. gov/cars/rules/rulings/tirepresfinal/index. html (2000)
2. Atiya, A.F., Parlos, A.G.: New results on recurrent network training: unifying the algorithms and accelerating convergence. Neural Networks, IEEE Transactions on 11(3), 697–709 (2000)
3. Bengio, Y., Simard, P., Frasconi, P.: Learning long-term dependencies with gradient descent is difficult. Neural Networks, IEEE Transactions on 5(2), 157–166 (1994)
4. Bevly, D.M., Sheridan, R., Gerdes, J.C.: Integrating ins sensors with gps velocity measurements for continuous estimation of vehicle sideslip and tire cornering stiffness. In: American Control Conference, 2001. Proceedings of the 2001. vol. 1, pp. 25–30. IEEE (2001)

5. Connor, J., Atlas, L.: Recurrent neural networks and time series prediction. In: Neural Networks, 1991., IJCNN-91-Seattle International Joint Conference on. vol. 1, pp. 301–306. IEEE (1991)
6. Consortium, A., et al.: Intelligent tyre systems-state of the art and potential technologies. APOLLO Deliverable D7 for Project IST-2001-34372. Also available at http://www. vtt. fi/apollo., Technical Research Centre of Finland (VTT) (2003)
7. Dakhlallah, J., Glaser, S., Mammar, S., Sebsadji, Y.: Tire-road forces estimation using extended kalman filter and sideslip angle evaluation. In: American Control Conference, 2008. pp. 4597–4602. IEEE (2008)
8. De Jesús, O., Horn, J.M., Hagan, M.T.: Analysis of recurrent network training and suggestions for improvements. In: Neural Networks, 2001. Proceedings. IJCNN'01. International Joint Conference on. vol. 4, pp. 2632–2637. IEEE (2001)
9. Duchanoy, C.a.: Desarrollo de un modelo dinámico integral de un vehículo todo terreno con 6 subsistemas, su validación y estudio de maniobrabilidad y confort. Master's thesis, Centro de Investigación en Computación (2012)
10. Fischer, R.E., Tadic-Galeb, B., Yoder, P.R., Galeb, R.: Optical system design. Citeseer (2000)
11. Fortuna, L., Graziani, S., Rizzo, A., Xibilia, M.G.: Soft sensors for monitoring and control of industrial processes. Springer Science & Business Media (2007)
12. Ghaffari, A., Khodayari, A., Arvin, S., Alimardani, F.: An anfis design for prediction of future state of a vehicle in lane change behavior. In: Control System, Computing and Engineering (ICCSCE), 2011 IEEE International Conference on. pp. 156–161. IEEE (2011)
13. Hagan, M.T., Demuth, H.B., Beale, M.H., De Jesús, O.: Neural network design, vol. 20. PWS publishing company Boston (1996)
14. Hallonborg, U.: Super ellipse as tyre-ground contact area. Journal of Terramechanics 33(3), 125–132 (1996)
15. Hochreiter, S., Bengio, Y., Frasconi, P., Schmidhuber, J.: Gradient flow in recurrent nets: the difficulty of learning long-term dependencies (2001)
16. Horn, J., De Jesús, O., Hagan, M.T.: Spurious valleys in the error surface of recurrent networks—analysis and avoidance. Neural Networks, IEEE Transactions on 20(4), 686–700 (2009)
17. Hornik, K., Stinchcombe, M., White, H.: Multilayer feedforward networks are universal approximators. Neural networks 2(5), 359–366 (1989)
18. hpi.racing: Baja 5sc ss (2016), `http://www.hpiracing.com/en/kit/105734`
19. Ilonen, J., Kamarainen, J.K., Lampinen, J.: Differential evolution training algorithm for feed-forward neural networks. Neural Processing Letters 17(1), 93–105 (2003)
20. Keck, M.: A new approach of a piezoelectric vibration-based power generator to supply next generation tire sensor systems. In: Proceedings of IEEE Sensors. pp. 1299–1302 (2007)
21. Li, L., Wang, F.Y.: Advanced motion control and sensing for intelligent vehicles. Springer Science & Business Media (2007)
22. Makki, N., Pop-Iliev, R.: Piezoelectric power generation for sensor applications: design of a battery-less wireless tire pressure sensor. In: SPIE Microtechnologies. pp. 806618–806618. International Society for Optics and Photonics (2011)
23. Matsuzaki, R., Todoroki, A.: Passive wireless strain monitoring of actual tire using capacitance-resistance change and multiple spectral features. Sensors and Actuators, A: Physical 126(2), 277–286 (2006)
24. Matsuzaki, R., Todoroki, A.: Intelligent tires based on measurement of tire deformation. Journal of solid mechanics and Materials engineering 2(2), 269–280 (2008)

25. Matsuzaki, R., Todoroki, A.: Wireless monitoring of automobile tires for intelligent tires. Sensors 8(12), 8123–8138 (2008)
26. Miao, P., Liu, G., Zhang, D., Jiang, Y., Zhang, H., Zhou, H.: Sideslip angle soft-sensor based on neural network left inversion for multi-wheel independently driven electric vehicles. In: Neural Networks (IJCNN), 2014 International Joint Conference on. pp. 2171–2175. IEEE (2014)
27. Miller, S.L., Youngberg, B., Millie, A., Schweizer, P., Gerdes, J.C.: Calculating longitudinal wheel slip and tire parameters using gps velocity. In: American Control Conference, 2001. Proceedings of the 2001. vol. 3, pp. 1800–1805. IEEE (2001)
28. Phan, M.C., Beale, M.H., Hagan, M.T.: A procedure for training recurrent networks. In: The 2013 International Joint Conference on Neural Networks (IJCNN) (2013)
29. Slišković, D., Grbić, R., Hocenski, Ž.: Methods for plant data-based process modeling in soft-sensor development. AUTOMATIKA: časopis za automatiku, mjerenje, elektroniku, računarstvo i komunikacije 52(4), 306–318 (2012)
30. Suratgar, A.A., Tavakoli, M.B., Hoseinabadi, A.: Modified levenberg-marquardt method for neural networks training. World Acad Sci Eng Technol 6, 46–48 (2005)
31. Toplar, H.: Experimental analysis of smart tires (2014)
32. Xia, K., Yang, Y.: Three-dimensional finite element modeling of tire/ground interaction. International journal for numerical and analytical methods in geomechanics 36(4), 498–516 (2012)
33. Zhang, W., Ding, N., Yu, G., Zhou, W.: Virtual sensors design in vehicle sideslip angle and velocity of the centre of gravity estimation. In: Electronic Measurement & Instruments, 2009. ICEMI'09. 9th International Conference on. pp. 3–652. IEEE (2009)

Acoplamiento molecular basado en ligando por Complejidad LMC

Mauricio Martínez M., Miguel González-Mendoza

Tecnológico de Monterrey, Estado de México,
México

A00964166@itesm.mx, mgonza@itesm.mx

Resumen El *Acoplamiento molecular* enfrenta problemas asociados al análisis de entidades de alta dimensionalidad y pocas muestras; es el caso de los efectos del fenómeno de *Maldición de dimensionalidad* que se presentan cuando se utilizan algoritmos de *Aprendizaje supervisado* y *Optimización matemática* para acoplar moléculas basados en técnicas basados en *ligando* (*LBVS-Ligand-Based Virtual Screening*). Se propone la utilización del concepto de *Complejidad LMC* como medida de relevancia, para identificar moléculas que por comparación con *Compuestos activos* puedan ser propuestos como *Candidatos a medicamento*. El objetivo es medir la similaridad entre vectores mediante *Complejidad LMC* , y ordenar las comparaciones hechas entre ellos con respecto a ésta característica, para descubrir las moléculas más parecidas a los *Compuestos activos*. Se diseñó un algoritmo de medición de similaridad por *Complejidad LMC*, que comparó dos grupos de vectores de alta dimensionalidad; un grupo de vectores *Candidatos a medicamento* contra un grupo de *Compuestos activos* o *Medicamentos*. Los resultados muestran que la aplicación de este concepto sobre los *Compuestos activos* es más informativa que la búsqueda individual de los mejores *Candidatos a medicamento*. Ya que el grado de similaridad global que mantienen con los *Candidatos a medicamento* permite distinguir que *Compuestos activos* son los mejores vectores con los que coincidiran en alto grado los vectores *Candidatos*. La identificación de vectores por ordenamiento evitó algunos de los efectos del fenómeno de *Maldición de dimensionalidad*.

Palabras clave: Acoplamiento molecular, compuesto activo, candidato a medicamento, complejidad LMC, maldición de dimensionalidad.

Molecular Docking based on *ligand* by LMC Complexity

Abstract. *Molecular Docking* faces problems related to *Curse of dimensionality*, due to the fact that it analyzes data with high dimensionality and few samples. *LBVS-Ligand-Based Virtual Screening* conducts studies of docking among molecules using common attributes registered

in data bases. This branch of *Molecular Docking*, uses *Optimization methods* and *Machine learning* algorithms in order to discover molecules similar to known drugs and can be proposed as drug candidates. Such algorithms are affected by effects of *Curse of dimensionality*. It this paper we propose to use *LMC complexity measure* as similarity measurement among vectors in order to discover the best molecules to be drugs; and present an algorithm, which evaluates the similarity among vectors using this concept. The results suggest that application of this concept on *Drug Example* vectors; in order to classify other vectors as drugs candidates which is more informative than individually searching for vectors. Since the *Drug Examples* show a global similarity degree with drug candidate vectors. The aforementioned similarity degree makes it possible to deduce which elements of the *Drug Examples* show higher degree of similarity with drug candidates. Searching of vectors through individual comparison with *Drug Examples* was less efficient, because their classification is affected by the *Drug Examples* with a higher number of global discrepancies. Finally, the proposed algorithm avoids some of the *Curse of dimensionality* effects by using a ranking process where the best drug candidate vectors are those with the lowest complexity.

Keywords: Molecular docking, active compound, drug candidates, LMC complexity, curse of dimensionality.

1. Introducción

Las técnicas de *Acoplamiento molecular* (o *docking* por su denominación en inglés), buscan encontrar el mejor acoplamiento entre dos o más moléculas de tal forma que la afinidad entre ellas sea óptima, [6]. Tales métodos se aplican principalmente al diseño y descubrimiento de medicamentos, Química computacional, Biología molecular y Remediación ambiental entre otras áreas. Estos métodos se implementan mediante la modelación de uniones geométricas entre las moléculas como son: posición, flexibilidad y rotación, [22]. Lo anterior implica explorar el espacio de posibilidades de las características anteriores y evaluar la relevancia de las propiedades que se busca obtener de las moléculas una vez acopladas, [22].

Virtual screening en *Acoplamiento molecular* agrupa un conjunto de procedimientos basados en algoritmos computacionales que identifican nuevos acoplamientos entre moléculas en base a la similaridad, relaciones de actividad e inactividad, propiedades físicas, químicas, estructurales, funcionales, etc. Tales características, entre muchas otras, son registradas para millones de compuestos en diferentes bases de datos, [15],[7],[11],[24]. La información registrada en ellas es representada con vectores de muy alta dimensionalidad, los cuáles son analizados con *Métodos de optimización* y *Aprendizaje de Máquina*, [22],[1].

La naturaleza de estos datos, provoca que los algoritmos empleados para su análisis presenten algunos de los efectos de *Maldición de dimensionalidad*; como son el *Fenómeno del espacio vacío*, *Hipervolumen de cubos y esferas*, *Hipervolumen de corona esférica* y *Concentración de normas y distancias*, [7,14].

En *Virtual screening* existen dos tendencias en la búsqueda de acoplamiento entre moléculas; Métodos basados en estructuras (*SBVS-Structured Based Virtual* Screening) y basados en *ligando* (*LBVS-Ligand-Based Virtual Screening*). Estos últimos emplean algoritmos como *Máquinas de soporte vectorial*, *Arboles de decisión*, *Redes neuronales*, etc. Cuyo objetivo es clasificar compuestos descritos por etiquetas que registran sus atributos y ordenarlos de acuerdo a la afinidad que manifiesten para interactuar con una *Molécula objetivo*, [12]. Los retos que enfrentan estos algoritmos son grandes montos de datos a procesar, y el descubrimiento de formas de discriminación óptimas entre *Compuestos activos* de *inactivos*, [3].

1.1. Trabajos previos

Los principales usos de algoritmos de *Aprendizaje de Máquina* en *LBVS-Ligand-Based Virtual Screening*, es la Clasificación y Búsqueda de similaridad entre compuestos, además de la Identificación de Patrones de similaridad entre ellos. Las *Máquinas de Soporte Vectorial* son utilizadas en las dos primeras actividades. La idea básica en estos algoritmos, es la identificación de un hiperplano de decisión que separe los vectores de datos más cercanos a él de forma óptima; con el fin de clasificarlos en función de este plano. Estos métodos tienen la desventaja de tener un alto costo computacional, además de necesitar un conjunto de datos de entrenamiento adecuado. Su desempeño baja ante la presencia de ruido y traslape de clases, [12].

Los *Arboles de decisión* tienen como objetivo en ésta área, asociar atributos específicos de las moléculas con alguna actividad o propiedad de interés relacionada con su acoplamiento. Estos métodos están basados en métricas de *Ganancia de información*, *Razón de ganancia de información* e *Indice de Divergencia Gini*. Se construyen los árboles, determinando la separación entre ramas mediante medidas de bifurcación basadas en las métricas anteriormente mencionadas. La construcción puede ser de arriba hacia abajo (*Top-Down*) o de abajo hacia arriba (*Bottom-Up*). Son modelos simples, de fácil interpretación y validación. Sin embargo, sufren de alta varianza, pues cambios pequeños en las mediciones, desencadenan un número alto de bifurcaciones. Su diseño depende del tamaño del conjunto de datos de entrenamiento y pueden sufrir de sobreajuste, [12].

Los *Clasificadores Bayesianos simples* (*Naive Bayes* por su denominación en inglés), determinan la probabilidad de la ocurrencia de un evento B dado que se presenta un evento A; lo que es el Principio del *Teorema de Bayes*. Se aplica en problemas donde se busca la probabilidad de que un compuesto representado por un vector de descriptores sea activo, dado que se conoce el compuesto activo A y un conjunto de compuestos inactivos Z de entrenamiento; , Ecuación 1. Tienen un alto costo computacional cuando la dependencia condicional entre variables es alta, [12]:

$$p(C_A|Z) = \frac{p(C_A)p(Z|C_A)}{p(Z)}. \tag{1}$$

K-nearest neighbors es aplicado en la clasificación de compuestos, ordenamiento y predicción bajo una modalidad de regresión. Se basan en la utilización

de *Medidas de distancia* como las *Euclideanas*, *Manhattan*, *Mahalanobis*, etc. Dependen del tamaño de conjunto de datos de entrenamiento, [24].

1.2. Planteamiento del Problema y preguntas de investigación

Un problema de acoplamiento molecular involucra el diseño de funciones de optimización y métodos de búsqueda eficientes que exploren el espacio de soluciones, [7], [17]. Estos métodos deben ser rápidos y tienen que descubrir atributos relevantes que satisfagan las restricciones impuestas para los acoplamientos buscados entre distintas moléculas, además de satisfacer la o las funciones objetivo propuestas, [6], [20].

En algunos problemas de acoplamiento, se pregunta qué compuestos o moléculas registrados en algunas bases de datos tienen la estructura apropiada para acoplarse a una molécula receptora, tal que el acoplamiento resultante, presente propiedades activas para ser medicamento. Se buscan entonces aquellos compuestos que globalmente tengan un alto grado de similaridad con medicamentos ya conocidos y que tengan alta probabilidad de vincularse a una molécula objetivo.

Los algoritmos dedicados a estas actividades trabajan bajo *Aprendizaje supervisado* tienen procesos de aprendizaje largos y requieren un número de instancias específico para ser entrenados, y validados; además de sufrir algunos de los efectos de *Maldición de dimensionlidad* mencionados anteriormente. En cambio, los métodos de *Aprendizaje no supervisado* son rápidos debido a que su implementación depende del uso de *mediciones de relevancia* aplicadas a los datos, pero tienden a perder precisión debido a su dependencia de *parámetros* y *formas de distribución estadística*, *Factores de escala*, presencia de *outliers*, datos incompletos, etc.

Medidas de relevencia basadas en *Entropía de información* ofrecen cierta independencia respecto a los inconvenientes anteriormente mencionados y requieren únicamente de la identificación en los datos, de los eventos o categorias simples que componen las entidades estudiadas; junto con la frecuencia con que se presentan en ellos, [18]. Uno de los conceptos poco explorados basados en *Entropía de información*, es el de *Complejidad*; término que describe la medida de *Desorden* u *Orden* presente en un conjunto de datos, dada la razón que existe entre las Entropías de información individuales de los distintos eventos reconocidos en ellos y la *Máxima entropía* que sustentan si tuvieran una frecuencia uniforme, [2]. Varios investigadores han propuesto diferentes definiciones para ella, es el caso de *Medición de complejidad* propuesto por Shiner et. al., el de *Complejidad LMC* planteado por Ricardo López-Ruiz et. al. y la *Complejidad de Kolmogorov* entre otras, [19], [13], [5].

Los dos primeros conceptos bajo el punto de vista de *Medidas de relevancia* pueden ser implementadas sobre distintos tipos de datos; lo que no es posible con la definición de Complejidad Kolmogorov, [16]. *Medición de complejidad* y *Complejidad LMC* están basadas en los conceptos de *Orden* y *Desorden*, [15]. *Medición de complejidad* es una cantidad adimensional y describe el comportamiento de los eventos simples de un fenómeno por el producto entre *Desorden*

y *Orden*; Entidades con bajas magnitudes de *desorden* y baja *complejidad* indican un comportamiento predecible o invariante; si tienen altas magnitudes de *desorden* y baja *complejidad*, describen entidades cuya frecuencia de eventos es uniforme entre ellos y por tanto impredecibles, [4]. Altas magnitudes de *Medición de complejidad* supone la presencia de patrones discernibles entre los distintos eventos que componen las entidades analizadas, Ecuación 2, 3, 4, 5 y 6.

$$S = \sum_{i=1}^{n} -P_i log_2 P_i, \tag{2}$$

$$S_{max} = Log_2 N, \tag{3}$$

$$\Delta \equiv S/S_{max}, \tag{4}$$

$$\Omega \equiv 1 - \Delta, \tag{5}$$

$$\Gamma_{\alpha\beta} \equiv \Delta^{\alpha} \Omega^{\beta}. \tag{6}$$

De la *Entropía de información* o de *Shannon*; expresada en la ecuación 2, se deriva el concepto de *Medición de complejidad*. El *Desorden*; denotado por Δ, se define como la razón que existe entre la *Entropía de información* y la *Máxima entropía*; Ecuaciones 4, 2 y 3 respectivamente. Mientras que el *Orden* u Ω es la diferencia entre la unidad y Δ; Ecuación 5. *Orden* y *Desorden* son medidas complementarias. Finalmente, la *Medición de complejidad* o $\Gamma_{\alpha\beta}$; Ecuación 6, se define como el producto de *Orden* y *Desorden* para $\alpha = 1$ y $\beta = 1$ en su expresión más sencilla.

La *Complejidad LMC*, conceptualmente es expresada en términos de *Entropía de información* y *Desequilibrio*. Esta definición de Complejidad, es la ponderación de la *Entropía de información* con la distancia que mantiene la probabilidad simple de cada uno de los eventos que componen una entidad, respecto al inverso del número de ocurrencias total del espacio de eventos; Ecuación 7. En este caso, las mínimas magnitudes de complejidad se presentan cuando el número de eventos en una entidad es único, o existen tantos eventos distintos como datos se tienen de la entidad analizada. El caso de evento único, implica una entropía de magnitud cero y se denomina *Estado de cristal* debido a la predicibilidad del fenómeno. Cuando todos los datos de la entidad implican eventos distintos,el *Desequilibrio* es cero y la entropía adquiere su máxima magnitud, lo cuál implica un *Estado de gas*. Estas cualidades son adecuadas para comparar compuestos por similitud, [13], [4]:

$$C = H * D = -(K) * (\sum_{i=1}^{N} P_i * Log_2 P_i) * (\sum_{i=1}^{N} (P_i - \frac{1}{N})^2). \tag{7}$$

Lo anterior da origen a las siguientes preguntas: ¿Es posible plantear un problema de acoplamiento de moléculas en función del concepto de Complejidad?, ¿La sencillez de los cálculos pueden facilitar la búsqueda de compuestos o moléculas adecuados para acoplar con una molécula objetivo?, ¿Puede ser planteada una función objetivo para evaluar el grado de similaridad mediante el concepto de complejidad?.

1.3. Hipótesis y objetivos

La alta dimensionalidad de vectores para un problema de acoplamiento de moléculas basado en ligando (*LBVS-Ligand-Based Virtual Screening*) puede ser planteado en términos simples e interpretable de acuerdo al concepto de *Complejidad LMC*.

El objetivo entonces es diseñar un algoritmo que destaque la similaridad entre compuestos por medio de la *Complejidad LMC*. Planteando dos eventos simples para la comparación; *Acoplamiento* y *Desacoplamiento* entre atributos de compuestos. Todo lo anterior evitando procedimientos de exploración onerosos del espacio de soluciones y el diseño de funciones objetivo complicadas.

Se propone entonces ponderar el número de acoplamientos de un compuesto con una molécula objetivo o compuesto activo, por los atributos comunes en los que coinciden; o su desacoplamiento cuando tienen estados opuestos para un mismo atributo. La valoración de la afinidad entre ellos, dependerá de la Complejidad LMC que se desprende de la frecuencia entre acoplamientos y desacoplamientos. Un *Ordenamiento ascendente* de las magnitudes calculadas de *Complejidad LMC* para las distintas comparaciones entre *vectores no clasificados* y los distintos compuestos activos permitirán distinguir que vectores tienen el grado más alto de similaridad con ellos al corresponder con valores bajos de *Complejidad LMC* o de disimilitud para valores altos.

2. Materiales y métodos

DuPont Pharmaceuticals liberó para KDD Cup 2001 competition un conjunto de datos que comprenden 1908 compuestos[1] que acoplan con la molécula objetivo Trombina, constituidos de 139,351 atributos con representación binaria, [10] . Tales atributos se consideran activos cuando la posición que les representa dentro de un vector-compuesto tiene un cero, e inactivos cuando tienen un uno.

Estos vectores se dividen en dos conjuntos: 42 vectores considerados como *Medicamentos* o *Compuestos activos* y 1886 vectores que deben postularse como *Candidatos a medicamento* dada la similaridad que presenten con los compuestos activos. La tarea consiste en determinar cuáles son los mejores una vez que son comparados; para lo anterior se empleará el algoritmo 1 que utiliza *Complejidad LMC* para medir la similaridad entre vectores.

El algoritmo de construye en dos secciones: una de preprocesamiento y otra de cálculo de similaridad por *Complejidad LMC*. El preprocesamiento tiene como objetivos, identificar entre los vectores; *vectores activos* y *Candidatos a medicamento*, aquellos cuyas componentes en su totalidad son ceros; para posteriormente almacenar en un arreglo los números de vectores que tienen ésta característica. Sobre los vectores restantes se realiza una búsqueda de las posiciones en cada vector que dispongan de unos. Una Lista de listas recibirá

[1] Agradecemos a DuPont Pharmaceuticals Research Laboratories y KDD Cup 2001 por la disposición de este conjunto de datos mediante UCI Machine Learning Repository

las etiquetas de posición de los unos o una etiqueta de cero si el vector está compuesto únicamente por ceros. Habrá tantas listas como vectores tenga el conjunto de datos.

Una vez llevado a cabo lo anterior se calcula la similaridad por *Complejidad LMC* entre los *Candidatos a medicamento* y *vectores activos*, utilizando el arreglo que registró los vectores con ceros y la lista de listas. El proceso realizará tantas comparaciones como combinaciones posibles existan entre los dos conjuntos de vectores establecidos. Los eventos básicos de cada comparación serán los de *Acoplamiento* y *No Acoplamiento*. En el primer caso, cada acoplamiento puede ser por coincidencia por cero o por uno, el desacoplamiento es la diferencia de estados para un mismo atributo entre dos vectores.

El conteo de la frecuencia de los eventos anteriormente es la base del cálculo de *Complejidad LMC*. Las excepción considerada en este proceso, es cuando se comparan dos vectores compuestos solo de ceros; en este caso se asigna un valor cero a la complejidad pues ambos vectores son iguales. Cuando estos son distintos, se utiliza la *diferencia simétrica de conjuntos* con las etiquetas guardadas para cada lista correspondiente a los vectores comparados. La cardinalidad de ésta diferencia constituye el número de desacoplamientos, y cuando ésta cantidad se resta a la dimensionalidad de los vectores tenemos los acoplamientos. Por cada cálculo de similaridad por *Complejidad LMC* entre comparaciones se registra en una tabla los datos de los *Números de vector* comparados, *Frecuencia de Acoplamientos*, *No acoplamientos* y la *Magnitud de similaridad* por *Complejidad LMC*. Finalmente, se llevaran a cabo distintos ordenamientos sobre ésta para descubrir los mejores acoplamientos entre vectores.

El grado más alto de similaridad entre vectores serán aquellas comparaciones que tengan valores de *Complejidad LMC* muy cercanos a cero; por tanto el ordenamiento de las comparaciones por ésta característica será ascendente. Teniendo los mejores resultados al inicio del ordenamiento y los peores al final.

3. Resultados

Los primeros resultados mostraron la existencia de 593 vectores cuyos componentes están constituidos por ceros en su totalidad. De ellos, 2 pertenecen al grupo de *Compuestos activos*, y el resto a *Candidatos a medicamento*. Esto resultó en la presencia de 1182 valores de complejidad LMC cero, producto de las distintas comparaciones que hubo entre estos vectores. Un análisis por cuartiles de similaridad entre los vectores *Candidatos a medicamento* y cada uno de los *Compuestos activos* se muestra en la Figura 1.

Puede observarse que los *Compuestos activos* con bajas magnitudes de *Complejidad LMC* en su comparación con vectores *Candidatos a medicamento*, mantienen una dispersión alta con ellos. De forma opuesta, si la magnitud de la complejidad es alta, la dispersión de los vectores *Candidatos a medicamento* es baja con respecto al *Compuesto activo*.

La identificación individual de vectores *Candidatos a medicamento* con la más baja magnitud de *Complejidad LMC* con cada uno de los 42 *Compuestos activos*

Entrada: $Comp[1 \ldots N][1 \ldots M]$
Salida: $TabAcopPrHlmc[M][Reng_{Inact}, Reng_{Act}, Acoplan, NoAcoplan, Hlmc]$

inicio
  $i \leftarrow 0,\ j \leftarrow 0,\ k \leftarrow 0,\ V^{(0)}[N]$
  $RengListUnos[[N]] = RengListUnos[[1 \ldots L_1] \ldots [1 \ldots L_N]] \quad 1 \leq L_i \leq N$
  $Reng_{Inact},\ Reng_{Act},\ Acoplan,\ NoAcoplan,\ Hlmc$
  $TabAcopPrHlmc[M][Reng_{Inact}, Reng_{Act}, Acop, NoAcop, Hlmc]$
  para cada $i \in N$ **hacer**
    si $Comp[i][1 \ldots M] == \overrightarrow{0}$ **entonces**
      $V^{(0)}[j] \leftarrow i$
      $j \leftarrow j + 1$
    fin
  fin
  para cada $i \in N$ **hacer**
    si $i \in V^{(0)}$ **entonces**
      Añade$(RengListUnos[[L_i]], 0)$
    en otro caso
      para cada $j \in M$ **hacer**
        si $Comp[i][j] == 1$ **entonces**
          Añade$(RengListUnos[[L_i]], j)$
        fin
      fin
    fin
  fin
  para todo $Reng_{Inact} \in Indices(Comp_i != "Activo")$ **hacer**
    para todo $Reng_{Act} \in Indices(Comp_i == "Activo")$ **hacer**
      si $Reng_{Inact} \in V^{(0)} \vee Reng_{Act} \in V^{(0)}$ **entonces**
        si $Reng_{Inact} \in V^{(0)} \wedge Reng_{Act} \in V^{(0)}$ **entonces**
          $Acoplan = N$
        en otro caso
          si $Reng_{Inact} \in V^{(0)}$ **entonces**
            $Acoplan \leftarrow (N - \mathbf{card}(S_{RengListUnos[[Reng_{Inact}]]}))$
            $NoAcoplan \leftarrow \mathbf{card}(S_{RengListUnos[[Reng_{Inact}]]})$
          en otro caso
            $Acoplan \leftarrow (N - \mathbf{card}(S_{RengListUnos[[Reng_{Act}]]})$
            $NoAcoplan \leftarrow \mathbf{card}(S_{RengListUnos[[Reng_{Act}]]})$
          fin
        fin
      en otro caso
        $NoAcoplan \leftarrow$
        $\mathbf{card}(S_{RengListUnos[[Reng_{Inact}]]} \triangle S_{RengListUnos[[Reng_{Act}]]})$
        $Acoplan \leftarrow (N - NoAcoplan)$
      fin
      si $Acoplan == N$ **entonces**
        $Hlmc \leftarrow 0$
      en otro caso
        $Hlmc \leftarrow (\sum_{i=1}^{N} P_i * Log_2 P_i) * (\sum_{i=1}^{N} (P_i - \frac{1}{N})^2) \Big|_{i \in \{Acoplan, NoAcoplan\}}$
      fin
      $TabAcopPrHlmc_{k+1} \leftarrow c(Reng_{Inact}, Reng_{Act}, Acoplan, NoAcoplan, Hlmc)$
    fin
  fin
  return(TabAcopPrHlmc)
fin

Algoritmo 1: Complejidad LMC H*D

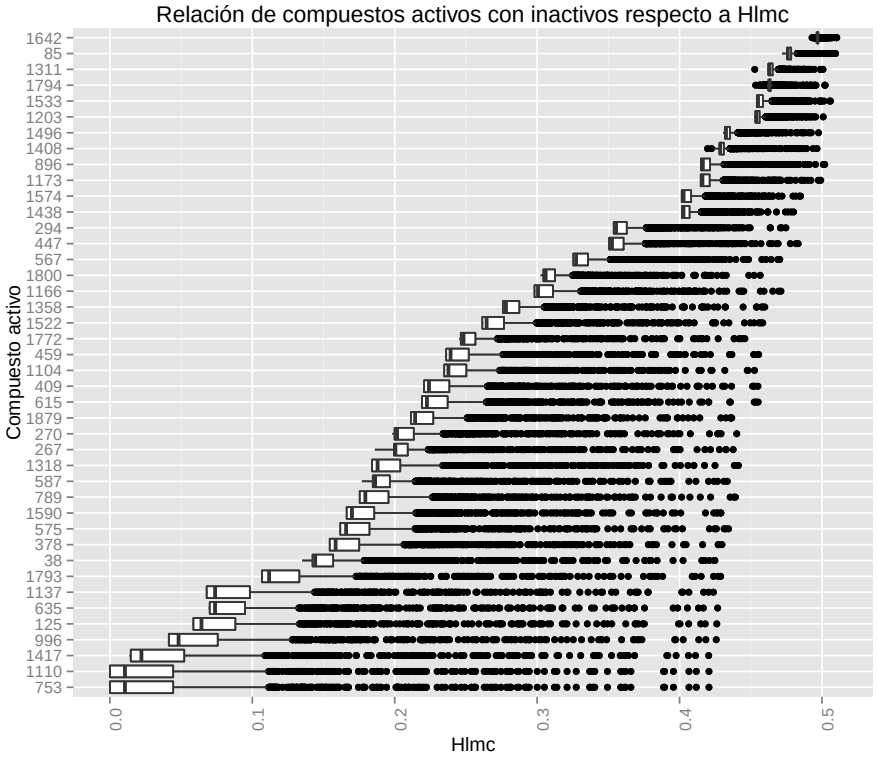


Fig. 1. Similaridad de medicamentos con vectores inactivos por *Complejidad LMC*

muestran que mantienen rangos de complejidad muy amplios en su análsis con boxplots. El ordenamiento de los *Candidatos a medicamento* bajo este criterio muestra valores de complejidad que van desde $3.648\text{x}10^{-4}$ hasta 0.51. La tabla 1 muestra los *Candidatos a medicamento* con más baja complejidad y sus valores máximos y mínimos al ser comparados con los *Compuestos activos*; así como el número de *Compuestos activos* con el que coincidieron bajo esta consideración.

De igual forma fueron identificados los *Candidatos a medicamento* que muestran los más altos niveles de Complejidad LMC, los resultados se muestran en la tabla 2. Se observa, que el número de *Compuestos activos* con los que coinciden es más alto con respecto a los compuestos que ofrecen baja complejidad, pero los grados de similaridad que sustentan son los más bajos.

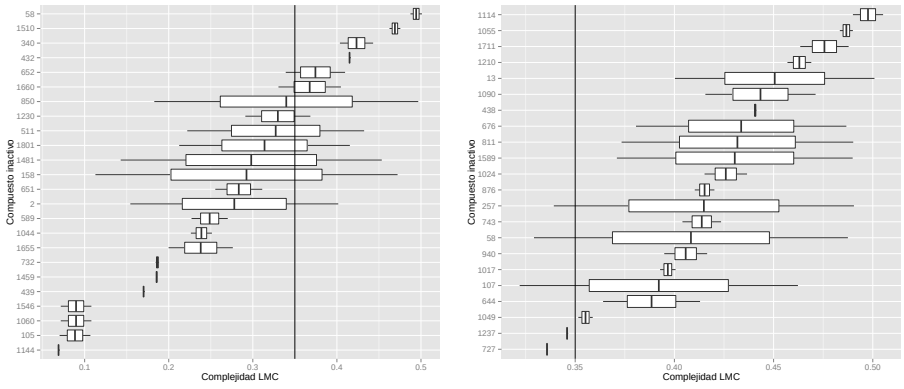
El análisis de similaridad de los *Candidatos a medicamento* con los *Compuestos activos* por boxplots, muestra un comportamiento mixto en la dispersión de estos últimos con respecto a los primeros. Aunque Los *Candidatos a medicamento* con baja complejidad, tienen niveles de dispersión más bajos comparados con los de alta, Figura 2.

Tabla 1. Compuestos Inactivos con baja *Complejidad LMC*

V_{Inact}	$C_{Max.}$	$C_{Min.}$	No. Med.
2	0.041	0.35	13
105	0.0003	0.0003	2
511	0.35	0.4	2
1060	0.18	0.41	3
1459	0.17	0.18	3

Tabla 2. Compuestos Inactivos con alta *Complejidad LMC*

V_{Inact}	$C_{Max.}$	$C_{Min.}$	No. Med.
13	0.051	0.49	4
438	0.47	0.42	19
1024	0.5	0.43	4
1090	0.5	0.42	12
1210	0.5	0.45	3



Vectores cercanos a complejidad cero Vectores con máxima complejidad

Fig. 2. Vectores inactivos

Para el primer recuadro de la figura 2, se observa la coincidencia en posición y ancho de los boxplots para los vectores 105,1060 y 1546. Al verificar el estado de sus atributos, se encontró que muy pocos están en estado inactivo; solo 3 para el vector 105 y de 32 para los últimos dos. Una observación similar se realizó para el segundo recuadro de la figura 2, donde se observa una proximidad alta entre los boxplots de los vectores 676, 811 y 1589; en ellos el número de atributos inactivos es alto, con 12598, 10791 y 12147 respectivamente.

4. Discusión

El acoplamiento de moléculas basados en *ligando* (*LBVS-Ligand-Based Virtual Screening*) utilizando el concepto de *Complejidad LMC* destaca el grado de similaridad globlal que mantienen los *Compuestos activos* con los *Candidatos a medicamento*. Esta característica, ofrece la ventaja de identificar las mejores ejemplos de aprendizaje entre los *vectores activos* para descubrir *Candidatos a medicamento* por similaridad. Sin embargo, la *Identificación individual* por complejidad de los mejores *Candidatos*, tienen el inconveniente de ser afectados por las comparaciones que tienen con los *Compuestos activos* menos eficientes en términos de similitud. Una excepción a las observaciones anteriores, fueron aquellas comparaciones entre vectores que presentaron magnitudes de *Complejidad LMC* cero. Lo que implicó acoples perfectos entre ellos; pudo haber sido interesante si los valores binarios de sus componentes hubieran sido mixtos, pero al revisarlos, se encontró que todos sus componentes tenían ceros.

El Algoritmo propuesto evita algunas de las debilidades asociadas a los métodos utilizados en el acoplamiento de moléculas basados en ligando. La etapa de preprocesamiento sortea la *Exploración del espacio de soluciones* presente en los *Métodos de optimización*, y en el caso de *Aprendizaje de Máquina*, no existe la necesidad de conjuntos de datos balanceados y representativos para procesos de aprendizaje. Los efectos de *Maldición de dimensionalidad* presentes durante el análisis de un número reducido de vectores con alta dimensionalidad, fue compensado al calcular su similaridad mediante el concepto de *Complejidad LMC*. La identificación de eventos simples como los de *Acoplamiento* y *No Acoplamiento* en la comparación de vectores permite la implementación de este concepto en un algoritmo de acoplamiento entre moléculas.

El tiempo de computo se reduce considerablemente al comparar solo las etiquetas de posición de las componentes de los vectores que contienen unos. La medición de similaridad entre dos vectores se implementa utilizando la *diferencia simétrica de conjuntos* con las etiquetas, ya que ésta operación matemática muestra los elementos exclusivos de cada vector; mientras que los elementos restantes son comunes. La cardinalidad de la *diferencia simétrica* es el número de desacoplamientos entre los vectores, y las etiquetas complementarias a la *diferencia simétrica* más las posiciones con valores cero son los acoplamientos.

Finalmente, una característica que se observa en los experimentos es que la naturaleza binaria de la representación de los vectores, determina el costo computacional del algoritmo propuesto. Si la distribución en frecuencia de unos y ceros tiende a ser simétrica, las comparaciones por atributo aumentan y por consecuencia el tiempo de computo. El caso contrario, una distribución asimétrica de estos valores requiere menos tiempo de computo, porque solo se comparan las posiciones de menor frecuencia. Lo anterior implica encontrar vectores *Candidatos a medicamento* de muy baja similaridad con los *Compuestos activos* cuando la frecuencia de unos y ceros tiende a ser simétrica y de alta similaridad para el caso alterno.

5. Conclusiones

El planteamiento de un problema de *Acoplamiento molecular basado en ligando* mediante *Complejidad LMC* está determinado por la identificación de los eventos simples que componen la comparación de dos vectores. El problema planteado en este artículo de medición de similaridad entre vectores *Candidatos a medicamento* y *vectores activos* o ejemplos de medicamento requirió identificar sólo dos eventos; *Acoplamientos* y *No acoplamientos*.

El cálculo de la expresión matemática de *Complejidad LMC* utilizada como *Medida de relevancia* para medir el *Grado de similaridad* entre vectores no requiere la exploración de espacios de soluciones ni procesos de aprendizaje largos, por tanto su implementación en un algoritmo es sencilla y no requiere del diseño de una función objetivo. La evaluación de los resultados para identificar los *Candidatos a medicamento* con el más alto grado de similaridad con los *vectores activos* se realizó por ordenamiento.

la interpretación de la similaridad entre vectores mediante la magnitud de *Complejidad LMC* depende de la *Entropía de información* y el *Desequilibrio*. La *Entropía* determina una magnitud baja de complejidad cuando la frecuencia de alguno de los eventos (*Acoplamientos* o *No acoplamientos*) es preponderante con respecto a los demás. Altas frecuencias de *Acoplamientos*; determinadas por el número de etiquetas comunes en posición y contenido, representan grados de similaridad altos. Si la frecuencia de los eventos es uniforme se obtiene una *Entropía de información* máxima y por lo tanto alta complejidad, lo que significa que la similaridad de los vectores comparados es baja debido a que existe igual número de acoplamiento que de desacoplamientos.

Una complejidad baja por *Desequilibrio* no se presentó en éste análisis, debido a que el número de eventos es mínimo comparado con el número de componentes de los vectores. La interpretación de los resultados permitió deducir que la mejor aproximación de similaridad entre vectores es con respecto a los vectores activos.

los efectos de *Maldición de dimensionalidad* en el algoritmo propuesto son mínimos debido al empleo de ordenamiento por magnitud de complejidad de los distintos acomplamientos existentes entre *Candidatos a medicamento* y *vectores activos*; lo que evita también altos costos computacionales, [23]. Aúnque no es necesario un conjunto de datos entrenamiento como lo exigen los *Métodos de aprendizaje supervisado*, la precisión del algoritmo propuesto es dificíl de determinar, [9]; pues como *Medida de relevancia* la *Complejidad LMC* es un indicador del grado de predicibilidad u orden que sustentan la relación de similaridad entre dos vectores; no se considera un cálculo de error con respecto a un patrón. Aunque, si existe una apreciación global que permite distinguir los *vectores activos* que sustentan el máximo grado de similarida con los *Candidatos a medicamento*, [21]. Consideramos que el concepto de *Complejidad LMC* es una opción flexible en el análisis de *Acoplamiento molecular basado en ligando* y que puede combinar el conocimiento registrados en bases de datos de distintas moléculas junto con sus caracterśticas estructurales, [8].

Referencias

1. Christos A. Nicolaou1, N.B.: Multi-objective optimization methodsin drug design. Drug discovery today 30(20) (2013)
2. Crutchfield, J.P.: Between order and chaos. Nature Physics 8(1), 17–24 (2012)
3. Danishuddin, M., Khan, A.U.: Virtual screening strategies: A state of art to combat with multiple drug resistance strains. MOJ Proteomics Bioinform 2 (2015)
4. Feldman, D.P., Crutchfield, J.P.: Measures of statistical complexity: Why? Physics Letters A 238(4), 244–252 (1998)
5. Grünwald, P.D., Vitányi, P.M.: Kolmogorov complexity and information theory. with an interpretation in terms of questions and answers. Journal of Logic, Language and Information 12(4), 497–529 (2003)
6. Halperin, I., Ma, B., Wolfson, H., Nussinov, R.: Principles of docking: An overview of search algorithms and a guide to scoring functions. Proteins: Structure, Function, and Bioinformatics 47(4), 409–443 (2002)
7. Karthikeyan, M., Vyas, R.: Practical chemoinformatics. Springer (2014)
8. Klebe, G.: Virtual ligand screening: strategies, perspectives and limitations. Drug discovery today 11(13), 580–594 (2006)
9. Kurczab, R., Smusz, S., Bojarski, A.J.: Evaluation of different machine learning methods for ligand-based virtual 3(S-1), 41 (2011)
10. Laboratories, D.P.R.: Dorothea data set, https://archive.ics.uci.edu/ml/datasets/Dorothea
11. Lavecchia, A., Di Giovanni, C.: Virtual screening strategies in drug discovery: a critical review. Current medicinal chemistry 20(23), 2839–2860 (2013)
12. Lavecchia, A.: Machine-learning approaches in drug discovery: methods and applications. Drug discovery today 20(3), 318–331 (2015)
13. Lopez-Ruiz, R., Mancini, H., Calbet, X.: A statistical measure of complexity. arXiv preprint nlin/0205033 (2002)
14. Robert Clarke, Habtom W. Ressom, e.a.: The properties of high-dimensional data spaces: implications for exploring gene and protein expression data. Nature Reviews Cancer 8, 13 (2008)
15. Robert P. Sheridan, S.K.K.: Why do we need so many chemical similarity search methods? Drug Discovery Today 7(17), 903–911 (2002)
16. Seaward, L., Matwin, S.: Intrinsic plagiarism detection using complexity analysis. In: Proc. SEPLN. pp. 56–61 (2009)
17. Shan, S., Wang: Survey of modeling and optimization strategies to solve high-dimensional design problems with computationally-expensive black-box functions. Structural and Multidisciplinary Optimization 41(2), 219–241 (Mar 2010), http://dx.doi.org/10.1007/s00158-009-0420-2
18. Shannon, C.E.: A mathematical theory of communication. The Bell System Technical Journal 27, 10–12 (1948)
19. Shiner, J.S., Davison, M., Landsberg, P.T.: Simple measure for complexity. Physical review E 59(2), 1459 (1999)
20. Sousa, S., Ribeiro, A., Coimbra, J., Neves, R., Martins, S., Moorthy, N., Fernandes, P., Ramos, M.: Protein-ligand docking in the new millennium–a retrospective of 10 years in the field. Current medicinal chemistry 20(18), 2296–2314 (2013)
21. Tanrikulu, Y., Krüger, B., Proschak, E.: The holistic integration of virtual screening in drug discovery. Drug Discovery Today 18(7), 358–364 (2013)
22. Teodoro, M.L., Phillips, G.N.: Molecular docking: A problem with thousands of degrees of freedom. In: IEEE International Conference on Robotics and Automation. pp. 960–966 (2001)

23. Zhang, W., Ji, L., Chen, Y., Tang, K., Wang, H., Zhu, R., Jia, W., Cao, Z., Liu, Q.: When drug discovery meets web search: Learning to rank for ligand-based virtual screening. J. Cheminformatics 7, 5 (2015)
24. Zheng, M., Liu, Z., Yan, X., Ding, Q., Gu, Q., platform for ligand-based virtual screening using publicly databases. Molecular diversity 18(4), 829–840 (2014)

Impreso en los Talleres Gráficos
de la Dirección de Publicaciones
del Instituto Politécnico Nacional
Tresguerras 27, Centro Histórico, México, D.F.
septiembre de 2016
Printing 500 / Edición 500 ejemplares